UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

PROYECTO FIN DE CARRERA

Estimación de confianza en la búsqueda de palabras clave en sistemas basados en transcripción fonética

Miguel Bautista Lozano

Septiembre 2012

Estimación de confianza en la búsqueda de palabras clave en sistemas basados en transcripción fonética

AUTOR: Miguel Bautista Lozano
TUTOR: Javier Tejedor Noguerales

Dpto. de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Septiembre de 2012

Resumen

En este proyecto, tras estudiar la arquitectura de un sistema STD (Spoken Term Detection o Reconocimiento de Palabras Clave), se desarrolla una herramienta de búsqueda de términos dentro de lattices fonéticos, generados a través de HTK (basada en la búsqueda en lattices Sphynx).

También se identifican una nueva serie de características fonéticas que ayudarán a la mejora del reconocimiento final de un término y se implementan las herramientas para aislar, extraer, y presentar esas características.

En la parte experimental, se evalúa nuestro sistema sobre el corpus geográfico de voz leída en español (ALBAYZIN) y sobre una base de datos de voz conversacional, de reuniones, en inglés (AMI).

Finalmente, se analizan los resultados a través de Análisis de Regresión y a través de la herramienta de aprendizaje automático WEKA, y se presentan conclusiones.

***Palabras clave*:**
Detector de palabras clave, HTK (Hidden Markov Model Toolkit), WEKA (Waikato Environment for Knowledge Analysis), búsqueda en lattice, selección de características fonéticas.

Abstract

In this project, after studying the architecture of an STD (Spoken Term Detection) system, we develop a Search of Terms in phonetic lattices Tool for lattices generated by HTK (based on search over Sphynx lattices).

We also identify a new set of phonetic features that will help to improve the final term recognition and we implement the tools to isolate, extract, and show those features.

Experimental tests consist of an evaluation of our system over the geographical corpus of read-speech in Spanish (ALBAYZIN) and a database of conversational speech in English (AMI).

Finally, the results are analyzed through Regression Analysis and through the WEKA Machine Learning Tool, and conclusions are drawn.

Keywords:
Spoken Term Detection, HTK (Hidden Markov Model Toolkit), WEKA (Waikato Environment for Knowledge Analysis), lattice search, phonetic feature selection.

Agradecimientos

En primer lugar quiero dar las gracias a todo el personal de la Escuela Politécnica Superior de la Universidad Autónoma de Madrid y en especial a mi tutor, Javier Tejedor Noguerales, que siempre ha estado disponible para ayudarme en todo lo que he necesitado, al igual que Doroteo Torre Toledano.

También quiero dar las gracias al personal de administración del Servicio de Idiomas de la Universidad: a Rodolfo, Ana, Inés, Rosalía, a los que nunca olvidaré por darme la oportunidad de trabajar junto a ellos en lo que fue mi primera experiencia laboral detrás de un ordenador.

Y por supuesto, gracias a todos los compañeros de la Escuela que, casi sin quererlo, se convirtieron en amigos tras muchas horas de estudio y mucho tiempo compartido. Sin duda, todos ellos han sido fundamentales para hacer de estos años en la Universidad una experiencia inolvidable por el esfuerzo exigido pero también por las personas que he conocido.

Gracias a todos los amigos que me han apoyado en estos últimos años, a mis primos, tíos y a mis abuelos. A las primas chiquitinas Carlota, Sofía y Berta, por ser un soplo de aire fresco y hacernos olvidar los malos momentos.

No han sido fáciles estos últimos años, así que gracias a todos por estar ahí.

Para finalizar, gracias a mi madre y a mi hermano. Gracias Maribel, Diego, Estela, que sé que habéis sufrido conmigo (y que me habéis sufrido de cerca) y ahora es el momento de que os lo agradezca.

…y gracias a Juan Manuel, mi padre.

Miguel Bautista Lozano
Septiembre 2012

INDICE DE CONTENIDOS

1 Introducción 7
1.1 Motivación 7
1.2 Objetivos 8
1.3 Organización de la memoria 9
2 Estado del arte 10
2.1 Sonido y procesamiento humano del habla 10
2.1.1 Sonido 11
2.1.2 Características acústicas 12
2.1.3 Fono y fonema 12
2.1.4 Producción y percepción del habla 12
2.2 Reconocimiento de Voz 15
2.2.1 Fundamentos de HTK 16
2.2.2 Modelos Ocultos de Markov (HMM) 16
2.2.3 Sistema de Reconocimiento HTK 18
2.2.3.1 Extracción de características 19
2.2.3.2 Herramientas de entrenamiento 20
2.2.3.3 Reconocedor 22
2.3 Detectores de Palabra Clave. Spoken Term Detection 23
2.3.1 Detectores de Palabra Clave 23
2.3.1.1 Sistemas de reconocimiento de gran vocabulario 23
2.3.1.2 Sistemas basados en modelos de relleno 24
2.3.1.3 Sistemas basados en reconocedores de sub-unidades de palabra 25
2.3.2 Spoken Term Detection 26
2.3.2.1 Introducción 26
2.3.2.2 Detección y medida de la confianza 27
2.3.2.3 Decisor (Decision Maker) 28
2.3.2.4 Evaluación 29
3 Medios disponibles y Diseño 30
3.1 Medios disponibles 30
3.1.1 Base de datos ALBAYZIN (Español, corpus geográfico) 30
3.1.2 Base de datos AMI (Inglés) 31
3.1.3 Archivos de Pitch y Energía de ficheros de audio (PRAAT) 32
3.1.4 Hardware 32
3.1.5 Software 32
3.1.5.1 HTK (Hidden Markov Model Toolkit) 32
3.1.5.2 Software de búsqueda en lattices Sphinx 33
3.1.5.3 Herramienta de aprendizaje automático WEKA 34
3.2 Diseño 36
3.2.1 Detección de palabras/fonemas en lattices 36
3.2.2 Extracción de características iniciales 37
3.2.3 Evaluación y clasificación de características 38
4 Desarrollo 39
4.1 Modificación de la herramienta de Búsqueda en lattice para HTK (basado en SPHYNX) 39
4.1.1 Formato Lattice Sphynx 40
4.1.2 Formato Lattice HTK 41

4.1.3 Búsqueda en Lattice (HTK) 42
4.1.3.1 Input/Output 42
4.1.3.2 Pseudocódigo 44
4.2 Extracción de Características a Nivel de Fonema 45
4.2.1 Características identificadas y extraídas 45
4.2.2 Extracción del pitch y energía (a nivel de fonema) 50
4.2.3 Pseudocódigo 51
5 Pruebas y resultados 56
5.1 Búsqueda en Lattices 56
5.1.1 ALBAYZIN 56
5.1.2 AMI 58
5.2 Extracción de Características y Cálculo del Pitch y Energía 59
5.2.1 ALBAYZIN 59
5.2.2 AMI 59
5.3 Análisis de Resultados 60
5.3.1 ALBAYZIN 60
5.3.1.1 Análisis Incremental de la Varianza 60
5.3.1.2 Evaluación de Atributos utilizando WEKA 66
5.3.1.3 Clasificación a través de Árbol de Decisión 68
5.3.2 AMI 70
5.3.2.1 Análisis Incremental de la Varianza 70
5.3.2.2 Evaluación de Atributos utilizando WEKA 76
5.3.2.3 Clasificación a través de Árbol de Decisión 78
6 Conclusiones y trabajo futuro 80
7 Referencias 84
8 Glosario I
9 Anexos I
A.Listado de Fonemas Españoles II
B.Listado de Fonemas Ingleses III
C. Publicaciones en Congresos Internacionales IV
10 Presupuesto XIV
11 Pliego de Condiciones XV

INDICE DE FIGURAS

Figura 1. Esquema de la onda sinusoidal producida por la compresión y expansión de moléculas de aire. 11

Figura 2. Tracto Vocal 13

Figura 3. Percepción del Habla. El oído humano 14

Figura 4. Esquema del reconocedor de voz HTK [8] 16

Figura 5. Modelo de generación de Markov [8] 17

Figura 6. Extracción de características en el proceso de habla humana [8] 18

Figura 7. Esquema de un sistema ASR estándar [1] 19

Figura 8. Módulos de entrada y de parametrización del audio 20

Figura 9. Herramientas de entrenamiento. Construcción de HMM 21

Figura 10. Algoritmo de Viterbi para la búsqueda del camino con mayor probabilidad. [9] 22

Figura 11. Arquitectura STD estándar 26

Figura 12. Ejemplo de curva DET con los valores de ATWV y max-ATWV 29

Figura 13. Ejemplo de Preprocesamiento de características con WEKA 34

Figura 14. Ejemplo de visualización de características en WEKA 35

Figura 15. Arquitectura de un STD estándar 36

Figura 16. Lista de características. [4] 37

Figura 17. Lista de nuevas características 38

Figura 18. Ejemplo de Lattice Sphynx 40

Figura 19. Ejemplo de Lattice HTK 41

Figura 20. Extracto de Lista de lattices 42

Figura 21. Extracto de archivo de Vocabulario (ALBAYZIN) 42

Figura 22. Archivo de Silencios (ALBAYZIN) 42

Figura 23. Wordmap 42

Figura 24. Archivo .mlf a nivel de palabra 43

Figura 25. Archivo .mlf a nivel de fonema (nunca extraído anteriormente) 43

Figura 26. Archivo de características a nivel de fonema 43

Figura 27. Ejemplo de fichero de características fonéticas extraídas a partir de lattice 49

Figura 28. Archivo de características prosódicas obtenido utilizando el archivo .mlf a nivel de fonema y el archivo de pitch y energía extraído con Praat 50

Figura 29. Archivo .mlf ("master label file") a nivel de palabra 56

Figura 30. Archivo .mlf a nivel de fonema 57

Figura 31. Archivos .mlf a nivel de fonema y a nivel de palabra (AMI) 58

Figura 32. Ejemplo de fichero de características fonéticas extraídas a partir de lattice (ALBAYZIN) 59

Figura 33. Ejemplo de fichero de salida, tras el cálculo de las nuevas características prosódicas a nivel de fonema (ALBAYZIN) 59

Figura 34. Ejemplo de fichero de características fonéticas extraídas a partir de lattice (AMI) 59

Figura 35. Ejemplo de fichero de salida, tras el cálculo de las nuevas características prosódicas a nivel de fonema (AMI) 59

Figura 36. Porcentaje de ocurrencias correctamente clasificadas (ALBAYZIN) 68

Figura 37. Tasa de Error Absoluto de Clasificación (ALBAYZIN). Nos da la medida de las diferencias en promedio entre los valores pronosticados y los observados 68

Figura 38. Porcentaje de ocurrencias correctamente clasificadas para el conjunto de datos TEST.INV (AMI) 78

Figura 39. Porcentaje de ocurrencias correctamente clasificadas para el conjunto de datos TEST.OOV (AMI) 78

Figura 40. Tasa de Error Absoluto de Clasificación para el conjunto de TEST.INV (AMI). Nos da la medida de las diferencias en promedio entre los valores pronosticados y los observados 79

Figura 41. Tasa de Error Absoluto de Clasificación para el conjunto de TEST.OOV (AMI). Nos da la medida de las diferencias en promedio entre los valores pronosticados y los observados 79

INDICE DE TABLAS

Tabla 1. Especificaciones de los corpora de la Base de Datos ALBAYZIN [5] 31

Tabla 2. Análisis de Regresión Lineal. Los conjuntos de características se van añadiendo en el orden que maximiza la varianza explicada. Se muestra la varianza explicada en porcentaje y el incremento absoluto con respecto al último conjunto de características añadido. 60

Tabla 3. Lista completa de características 61

Tabla 4. Análisis Incremental de la Varianza para el conjunto de datos TRAIN de ALBAYZIN 62

Tabla 5. Análisis Incremental de la Varianza para el conjunto de datos DEV de ALBAYZIN 63

Tabla 6. Análisis Incremental de la Varianza para el conjunto de datos TEST de ALBAYZIN 64

Tabla 7. Evaluación de Características GainRatio en conjunto de datos TRAIN (ALBAYZIN) 67

Tabla 8. Análisis Incremental de la Varianza para el conjunto de datos TRAIN de AMI . 71

Tabla 9. Análisis Incremental de la Varianza para el conjunto de datos DEV de AMI 72

Tabla 10. Análisis Incremental de la Varianza para el conjunto de datos TEST.INV de AMI 73

Tabla 11. Análisis Incremental de la Varianza para el conjunto de datos TEST.OOV de AMI 74

Tabla 12. Evaluación de Características GainRatio en conjunto de datos TRAIN (AMI).. 76

Tabla 13. Comparativa entre ALBAYZIN y AMI de la Evaluación GainRatio 82

Tabla 14. Listado de fonemas españoles junto con una palabra de ejemplo II

Tabla 15. Listado de fonemas ingleses junto con una palabra de ejemplo III

INTRODUCCIÓN

1.1 Motivación

En los últimos años, la investigación en el reconocimiento automático del habla se ha desarrollado intensamente debido en gran parte a la evolución de las herramientas y algoritmos de procesamiento de señales.

En la actualidad, prácticamente cualquier operación que se quiera hacer interactuando con un ordenador, dispositivo móvil, etc., es propensa a hacerse a través de la voz y por tanto se hace necesario incorporar un sistema de reconocimiento de voz.

Algunas de las aplicaciones más representativas son:

Control por comandos: Interacciones con un computador a través de la voz, mediante instrucciones concretas (cada vez suelen ser sistemas más eficaces debido al vocabulario reducido que utilizan).

Sistemas de telefonía: Algunas centralitas telefónicas disponen de sistemas de reconocimiento automático de voz junto o como alternativa a la marcación por tonos. Se suelen incorporar en menús de navegación, para precisar mejor la atención al cliente recopilando datos personales, etc.

Dispositivos móviles: Para sacar el máximo partido a estos dispositivos de tamaño reducido (como los teléfonos móviles) se hace casi imprescindible incorporarles sistemas de reconocimiento de voz para introducir datos y explotar todas las posibilidades que ofrecen.

Sistemas para discapacitados: Para personas con problemas de audición (herramientas que convierten la voz en texto), invidentes o personas con problemas para pulsar botones de forma fluida, interactuar a través de la voz se convierte en clave para mejorar notablemente su calidad de vida.

La investigación en reconocimiento de voz nos está llevando progresivamente hacia la transcripción de fuentes de información cada vez más complejas. Esto ha inspirado el seguimiento de la investigación sobre la tecnología de búsqueda y navegación en contenidos de audio ("Spoken Term Detection") [3].

1.2 Objetivos

Suponiendo un reconocedor de voz ya listo y entrenado que devuelve unos lattices fonéticos o de palabra, el objetivo es encontrar y/o ajustar parámetros que permitan determinar si una hipótesis de ocurrencia de una palabra es un acierto o un error, de la manera más precisa posible, a través de la información contenida en dichos lattices u otra información adicional con la que decidir si aceptar o no esa palabra.

A cada hipótesis (ocurrencia) se le asocia una puntuación o "Score" que suele estar basada únicamente en el modelo acústico y de lenguaje del reconocedor [2], por lo que uno de los objetivos más importantes del proyecto será reasignar puntuación a estas hipótesis en función de información y características adicionales obtenidas bien del lattice de reconocimiento o bien de otras características del audio. Para ello, la información será procesada a través de sistemas de aprendizaje automático como los "Árboles de Decisión (CARTs)" que finalmente asignarán una puntuación para cada ocurrencia que reflejará la seguridad o confianza que se tiene en que esa palabra aparece en el audio.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

1. Introducción: motivación y objetivos del proyecto.

2. Estado del arte en reconocimiento de voz y búsqueda de palabras clave en documentos hablados, así como información sobre el sonido y la producción y percepción del habla.

3. Medios disponibles y diseño del proyecto: medios a utilizar (software, hardware, bases de datos), sistema y datos de partida y descripción de las mejoras a implementar.

4. Desarrollo: Descripción de las aplicaciones implementadas (nuevas herramientas de búsqueda en lattices y de extracción de características fonéticas).

5. Pruebas y resultados obtenidos tras ejecutar las nuevas aplicaciones contra diferentes bases de datos y análisis de los resultados obtenidos.

6. Conclusiones que consideramos interesantes tras analizar los resultados e indicaciones sobre posibles futuras líneas de investigación.

7. Referencias con las distintas fuentes de información utilizadas en el proyecto.

8. Glosario de términos.

9. Anexos: Incluye listado de fonemas españoles e ingleses y publicaciones internacionales durante el proyecto.

10. Presupuesto y pliego de condiciones del proyecto.

2

ESTADO DEL ARTE

2.1 Sonido y procesamiento humano del habla

El proceso del habla comienza con la producción de una onda de presión que se transmite por el aire. A través de la sucesión en un orden concreto (característico de cada idioma) de los sonidos producidos, se elabora el mensaje.

Podemos analizar el habla desde diferentes perspectivas. La acústica se ocupa de analizar las características físicas de la voz, la semántica trata el sentido de las palabras, etc. A la hora de desarrollar sistemas de reconocimiento de voz todas estas ciencias son muy útiles ya que aportan información sobre el mensaje.

2.1.1 Sonido

El sonido es un conjunto de ondas producidas por un cuerpo al vibrar, que crean una variación de presión en el medio que le rodea. Para que se genere un sonido es necesario que vibre alguna fuente. Las vibraciones pueden ser transmitidas a través de diversos medios, siendo los más comunes el aire y el agua. El sonido humanamente audible consiste en ondas sonoras que producen oscilaciones de la presión del aire, que son convertidas en ondas mecánicas en el oído humano y percibidas por el cerebro. La propagación del sonido es similar en los fluídos, donde el sonido toma la forma de fluctuaciones de presión. En los cuerpos sólidos la propagación del sonido involucra variaciones del estado tensional del medio.

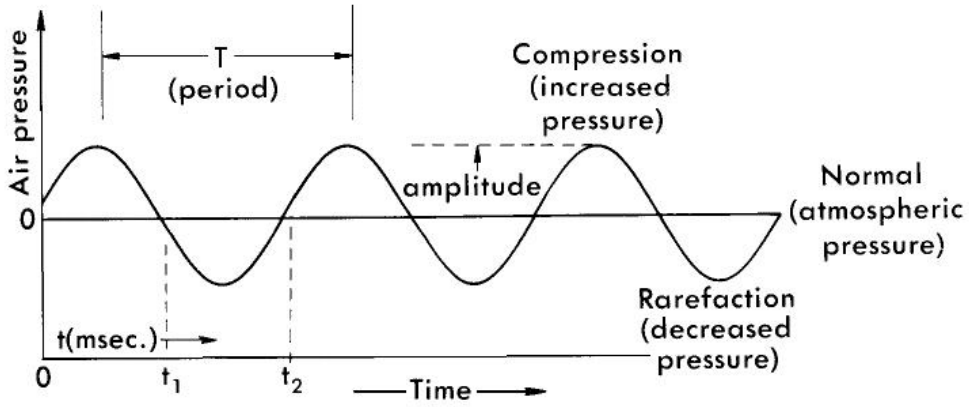


Figura 1. Esquema de la onda sinusoidal producida por la compresión y expansión de moléculas de aire.

El trabajo necesario para producir la energía que produce la compresión de moléculas de aire, se refleja en la cantidad de desplazamiento de las moléculas de aire desde su posición de reposo. Este desplazamiento se mide como la amplitud de un sonido. Debido al amplio rango de variación, es conveniente medir la amplitud del sonido en decibelios (dB) sobre una escala logarítmica.

El nivel de presión sonora o nivel de sonido (SPL, Sound Pressure Level) es una medida logarítmica de la presión acústica eficaz de un sonido en relación con un valor de referencia. El umbral de audición del oído humano está fijado en 0 dB y coincide con P_0 = 0.0002μbar para una señal de 1KHz, así que los sonidos con un nivel de presión de señal acústica por debajo de este umbral no serán audibles para el oído humano. Los sonidos con mayor valor de presión que el oído puede tolerar son en torno a los 120 dB, denominado umbral de dolor.

$$\text{SPL(dB)} = 20 \log_{10}\left(\frac{P}{P_0}\right) \qquad (1)$$

2.1.2 Características acústicas

La señal de voz es una sucesión de sonidos y silencios. Esta señal está formada por palabras, que a su vez se dividen en fonemas, los cuales son la unidad básica del habla y que unidos determinan los sonidos con los que construir el mensaje.

Los fonemas son unidades teóricas que utilizamos para analizar el nivel fonético de una lengua. Para estimar qué compone o no un fonema necesitamos que exista una función distintiva: son sonidos del habla que nos hacen diferenciar palabras en una lengua.

2.1.3 Fono y fonema

Los fonemas no son sonidos con entidad física, sino abstracciones mentales o abstracciones formales de los sonidos del habla. Un fonema puede ser representado por una familia o clase de equivalencia de sonidos (técnicamente denominados fonos), que los hablantes asocian a un sonido específico durante la producción o la percepción del habla. Así por ejemplo, en español el fonema /d/ [+ obstruyente, + alveolar, + sonoro] puede ser articulado como oclusiva [d] a principio de palabra o tras nasal o pausa larga, pero es pronunciado como aproximante [ð] entre vocales o entre vocal y líquida, así /dedo/ se pronuncia [deðo] donde el primer y tercer sonido difieren en el grado de obstrucción aunque son similares en una serie de rasgos (los propios del fonema).

Un sonido o fono está caracterizado por un conjunto de rasgos fonéticos y articulatorios en donde el número de dichos rasgos y la identificación de los mismos es tarea de la fonética. Un fono es cualquiera de las posibles realizaciones acústicas de un fonema. Por tanto, la fonética es la rama de la lingüística que estudia la producción y percepción de los sonidos de una lengua en sus manifestaciones físicas y sus principales ramas son: fonética experimental, fonética articulatoria, fonemática y fonética acústica.

2.1.4 Producción y percepción del habla

En el proceso de generación de la voz, el sonido inicial proviene de la vibración de las cuerdas vocales conocida como vibración glotal, es decir, el efecto sonoro se genera por la rápida apertura y cierre de las cuerdas vocales conjuntamente con el flujo de aire emitido desde los pulmones. Las cuerdas vocales comienzan a vibrar, produciéndose un sonido tonal, es decir periódico y cuya frecuencia varia en forma inversa al tamaño de las cuerdas. Este sonido es propio del hablante, es más agudo para el caso de mujeres y niños y carece de información lingüística.

Después de atravesar la glotis el sonido pasa a través de la cavidad supraglótica, que es la porción del aparato fonador que permite modificar el sonido dentro de márgenes muy amplios. Está conformado principalmente por tres cavidades, la cavidad oral, la cavidad labial y la cavidad nasal, correspondientes a la garganta, los labios y la nariz respectivamente. Estas cavidades constituyen resonadores acústicos, los cuales modifican los sonidos de acuerdo a la forma que adopten. La lengua y los labios permiten efectuar esta variación de manera voluntaria.

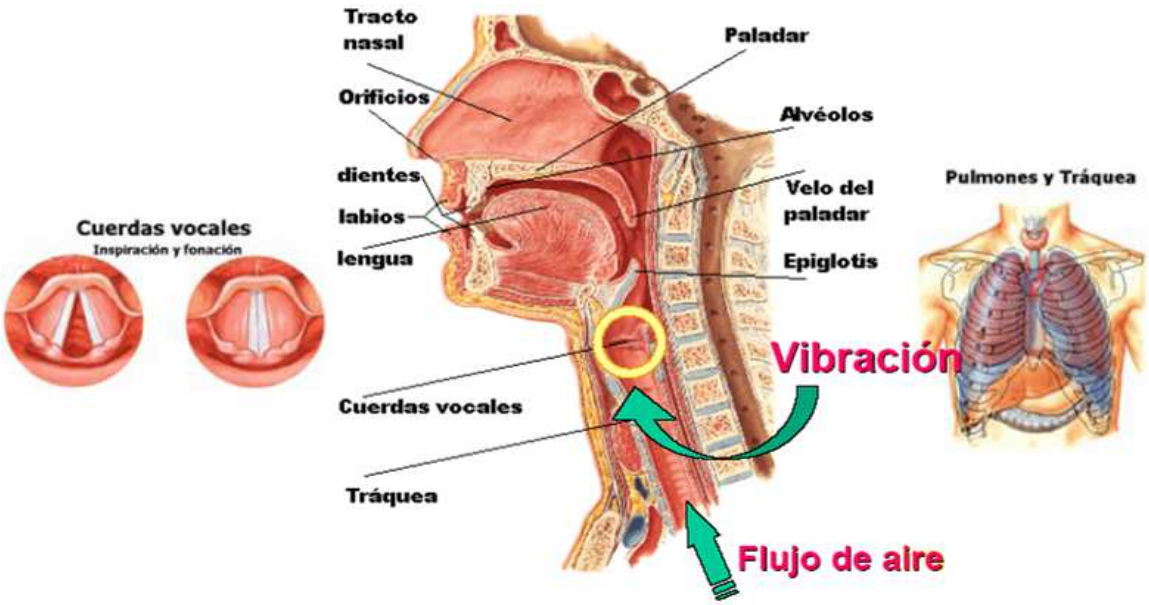


Figura 2. Tracto Vocal

Por otro lado, la capacidad de comprender el lenguaje oral se deriva del funcionamiento de un conjunto muy complejo de procesos perceptivos, cognitivos y lingüísticos que permiten al oyente recuperar el significado de un enunciado cuando lo oye.

La percepción puede verse como un proceso que une la onda acústica y su representación conceptual.

El objetivo es determinar cómo la señal acústica, que varía de manera continua, se convierte en una secuencia de unidades lingüísticas de forma que sea posible recuperar el mensaje. En el caso de que la señal de habla tenga mala calidad, el proceso de percepción continúa realizándose correctamente. Esto se debe a que el habla es una señal altamente estructurada y redundante de modo que las distorsiones no afectan a la inteligibilidad. La percepción también es posible porque el oyente tiene dos tipos de información disponibles, el contexto del habla (conocimiento pragmático) y el conocimiento de la lengua (sintaxis, semántica y fonología).

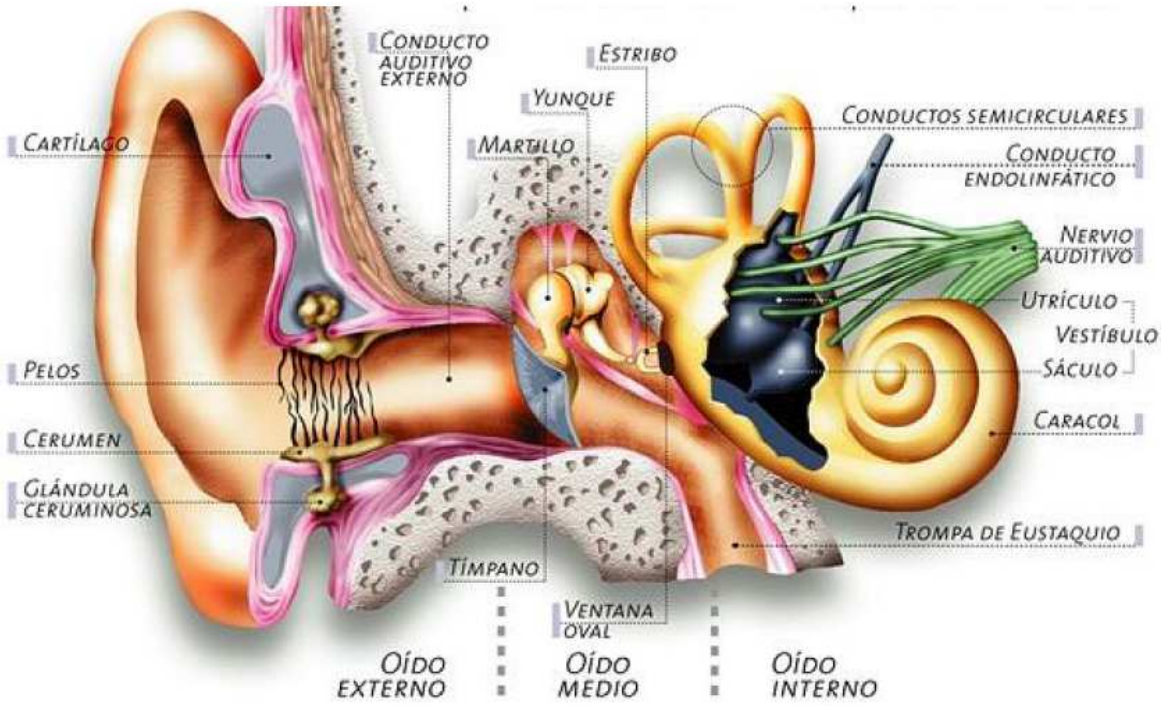


Figura 3. Percepción del Habla. El oído humano

2.2 Reconocimiento de Voz

El reconocimiento de voz es el proceso de conversión de una señal acústica en un conjunto de palabras (imitando el proceso de reconocimiento que lleva a cabo el receptor en la comunicación oral). El problema que se plantea en un sistema de reconocimiento de voz es el de hacer cooperar un conjunto de informaciones que provienen de diversas fuentes de conocimiento (acústica, fonética, fonológica, léxica, sintáctica, semántica, prosódica y pragmática), en presencia de ambigüedades, incertidumbres y errores inevitables para llegar a obtener una interpretación aceptable del mensaje acústico recibido.

- **Información acústica:** Obtenida parametrizando la señal analógica de entrada, principalmente en tiempo y en frecuencia.
- **Información fonética:** La información de características acústicas es traducida a una sucesión de fonemas.
- **Información fonológica:** Se analizan los fonemas que hacen que el contenido fonético de las palabras se modifique en una articulación rápida o por una sucesión de términos léxicos. Las variedades dialectales son también tratadas.
- **Información léxica:** Se identifican las palabras de la lengua en la que se produce la comunicación.
- **Información sintáctica:** Reglas gramaticales que permiten describir y analizar el lenguaje, y que relacionan las palabras reconocidas a nivel léxico.
- **Información semántica:** Trata el sentido de las palabras, buscando la comprensión del mensaje y eliminando las interpretaciones que no tengan sentido. Es el nivel de conocimiento de las palabras que da un diccionario de la lengua.
- **Información pragmática:** Analiza el sentido del mensaje recibido teniendo en cuenta el contexto de su aplicación. Reconoce la información que viene determinada por la situación en la que se produce la comunicación.
- **Información prosódica:** Información que el mensaje comunica mediante los modos de pronunciación: palabras pronunciadas con cierto nivel de insistencia para resaltarlas, fronteras entre grupos de palabras, naturaleza interrogativa o declarativa de una frase, etc. [18]

Un sistema de reconocimiento de voz es una herramienta computacional capaz de procesar la señal de voz emitida por el ser humano y reconocer la información contenida en ésta, convirtiéndola en texto o emitiendo órdenes que actúen sobre un proceso.

2.2.1 Fundamentos de HTK

El Hidden Markov Model Toolkit (HTK) es un conjunto de herramientas usado para la generación y manipulación de modelos de Markov. En un primer momento se desarrolló para construir modelos basados en el procesamiento de señales de habla y posteriormente se han ido encontrado muchas otras aplicaciones como la síntesis de voz o secuencias de ADN. [8]

HTK fue desarrollado originalmente por el departamento de Ingeniería de la Universidad de Cambridge, conocido como el grupo "the Speech Vision and Robotics". [17]

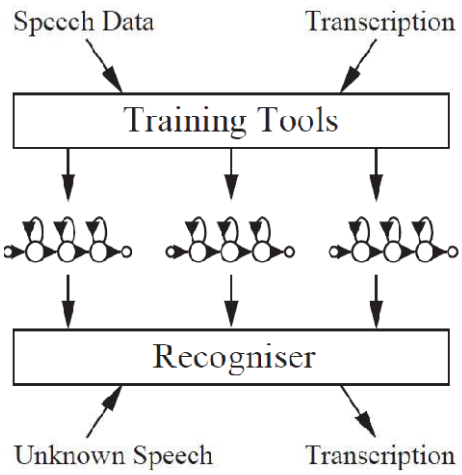


Figura 4. Esquema del reconocedor de voz HTK [8]

HTK se usa principalmente para el desarrollo de aplicaciones de procesamiento de señales de voz, en particular para reconocedores de voz. Como se puede ver en la figura 4, existen dos etapas principales en el proceso de reconocimiento. En primer lugar, las HTK Training Tools se utilizan para estimar los parámetros del HMM usando datos/señales de entrenamiento y sus transcripciones asociadas. En segundo lugar, se transcriben señales desconocidas usando las herramientas de reconocimiento. [8]

2.2.2 Modelos Ocultos de Markov (HMM)

Un HMM es una máquina de estados finita, en la que las observaciones son una función probabilística del estado. Esto quiere decir que el modelo es un proceso doblemente estocástico formado por un proceso estocástico oculto no observable directamente, que corresponde a las transiciones entre estados, además de un proceso estocástico observable cuya salida es la secuencia de vectores espectrales. [8]

Las observaciones son los vectores de parámetros acústicos y los estados suelen modelar eventos sonoros de menor duración que un fonema. Las densidades de probabilidad de cada estado, las probabilidades de transición y la secuencia de observaciones son parámetros conocidos del sistema, mientras que la secuencia de estados que el modelo de Markov ha seguido para generar esa secuencia de observaciones permanece desconocida para el usuario. [16]

Los elementos que definen un HMM [16] son:

• N: el número de estados del modelo, donde qt denota el estado en el instante de tiempo t. Los HMMs habitualmente están compuestos por N estados. Ni el estado 1 ni el N generan salida, y simplemente se usan para concatenar los diferentes modelos (por ejemplo, para reconocer palabras a partir de modelos de fonema o para reconocer frases a partir de modelos de palabra).
$S = \{s1, s2,...,sN\}$ (2)

• La dimensión del conjunto de observaciones distintas de salida M, es decir el tamaño del alfabeto.
$V = \{v1, v2,...,vM\}$ (3)

• La distribución de probabilidad de transición entre estados $A=\{aij\}$:
$aij = P(qt=sj \mid qt-1=si)\ 1 \leq i,j \leq N$ (4)

• La distribución de probabilidades de emisión de símbolos entre estados
$B = \{bj(k)\}$, (5)
$bj(Ok)=P(Ok \mid qt = sj)\ 1 \leq j \leq N,\ 1 \leq k \leq M$ (6)
, donde Ok es un símbolo perteneciente a V.

• Distribución del estado inicial $\pi = \{\pi i\}$:
$\pi i = P(q0 = si)\ 1 \leq i \leq N$ (7)

Con todo esto, un HMM se describe como $\lambda = \{A, B, \pi\}$. (8)

La técnica de HMM se usa en la actualidad en aquellos sistemas en los que el modelado tiene una dependencia del tiempo, como pueden ser los sistemas de reconocimiento fonético y del habla en general.

Una razón, por la que los HMMs son utilizados en el reconocimiento de fonemas, es que una señal de voz puede verse como una señal invariante a corto plazo (de unos 10 -20 milisegundos). La voz se podría interpretar así como un modelo de Markov para muchos procesos estocásticos (conocidos como estados). Otra razón por la que los HMMs son populares, es que pueden ser entrenados automáticamente, siendo factible realizar los cálculos en un tiempo razonable. [16]

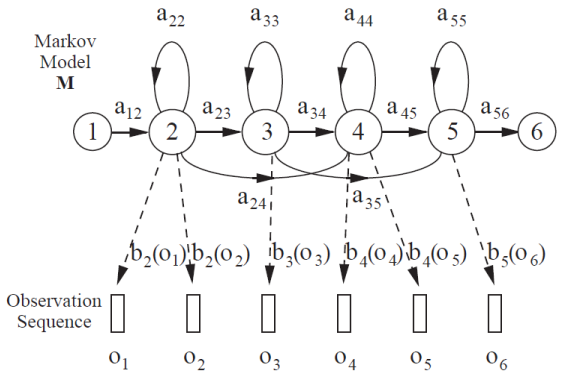


Figura 5. Modelo de generación de Markov [8]

2.2.3 Sistema de Reconocimiento HTK

Como ya adelantamos anteriormente, HTK es un toolkit principalmente diseñado para la construcción de herramientas de procesamiento de señales de voz, basadas en HMM, en particular para sistemas de reconocimiento de voz (ASR/RAV). Ofrece un conjunto de herramientas para llevar a cabo las funciones básicas relacionadas con el reconocimiento de voz:

- Proceso de extracción de características, a través del cual la señal acústica de entrada se transforma en una secuencia de vectores (Mel-Frequency Cepstral Coefficient (MFCC), Perceptual Linear Predictive (PLP), Linear Predictive Coding Coefficient (LPCC), etc, utilizados durante el entrenamiento de los modelos acústicos y el proceso de reconocimiento (decodificación).

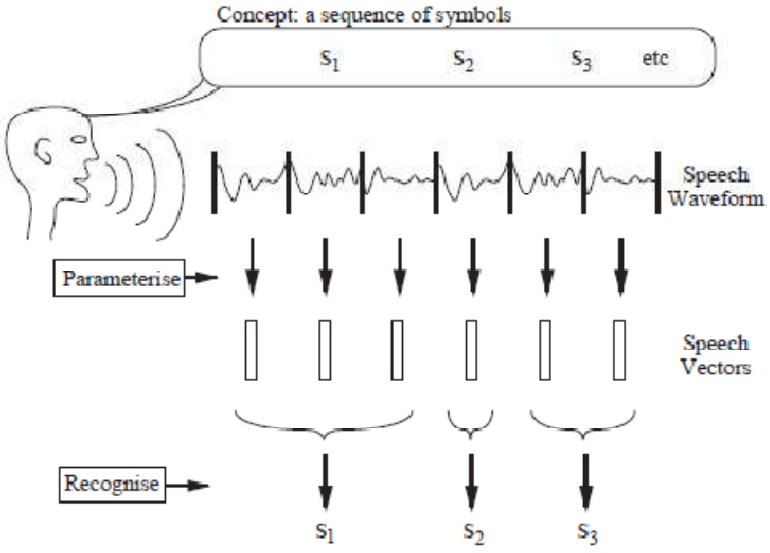


Figura 6. Extracción de características en el proceso de habla humana [8]

- Entrenamiento de los modelos acústicos, que consisten en la construcción de los HMMs utilizando el algoritmo de Baum-Welch, a partir de los vectores extraídos previamente.

- Finalmente, el proceso de reconocimiento a través del cual, utilizando el algoritmo de Viterbi, la señal de entrada es transformada en una cadena compuesta por las unidades requeridas (fonemas, palabras, etc).

Además, el toolkit también proporciona componentes para construir los modelos de lenguaje (LMs) que se utilizarán durante el proceso de reconocimiento. El conjunto de unidades a reconocer está definido en el lexicon del ASR y normalmente consiste en fonemas, grafemas, sílabas o palabras. La Figura 7 representa el esquema típico de un sistema ASR.

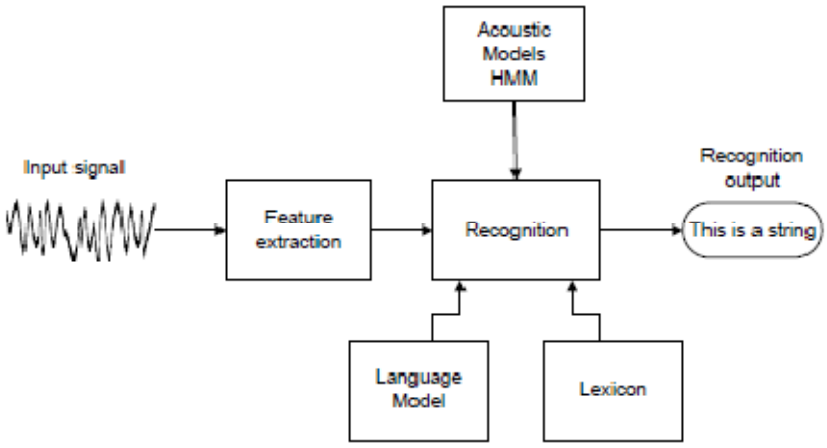


Figura 7. Esquema de un sistema ASR estándar [1]

2.2.3.1 Extracción de características

Durante esta primera etapa, se extraen las características acústicas que utilizará posteriormente el reconocedor. Para ello, se divide la señal de voz en una colección de segmentos y posteriormente, se obtiene una representación de las características acústicas más distintivas para cada segmento.

Con estas características, se construye un conjunto de vectores que constituyen la entrada al siguiente módulo. Una de las representaciones más usadas son los coeficientes Linear Predictive Coding (LPC) y los coeficientes Mel-Frecuency Cepstrum Coefficients (MFCC).

En HTK, el módulo HWave se encarga de recibir el archivo de audio. HAudio puede recibir una entrada directamente desde un dispositivo de audio. El módulo HParm se encarga de parametrizar la señal de entrada y generar los vectores usando los procedimientos y parámetros definidos en el módulo HSigP. HVQ añade índices VQ a los vectores obtenidos. [8]

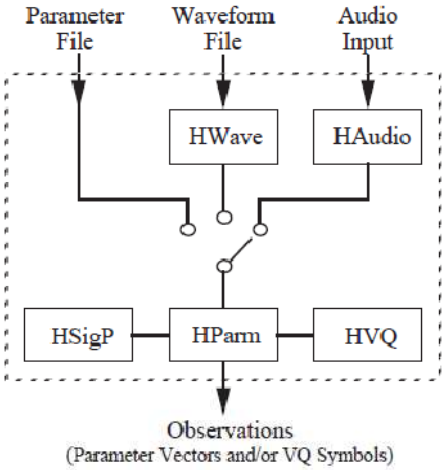


Figura 8. Módulos de entrada y de parametrización del audio

2.2.3.2 Herramientas de entrenamiento

El proceso de entrenamiento tiene lugar en distintas etapas, como se describe en la figura 9. Primero, se tiene que crear un conjunto inicial de modelos. Si tenemos datos de audio con alguna información identificada (ejemplo: identificación de los fonemas sobre la señal de audio), estos se pueden utilizar como datos de referencia. En este caso, las herramientas HInit y HRest inician un entrenamiento de palabras aisladas usando estos datos de referencia. Cada uno de los HMMs necesarios se genera individualmente. HInit lee todos los datos de referencia y localiza todos los ejemplos del fonema. Después los procesa iterativamente con un conjunto inicial de parámetros, utilizando un procedimiento "segmental k-means". [8]

En el primer ciclo, los datos de entrenamiento son segmentados uniformemente. Cada estado del modelo se ajusta a los segmentos de datos correspondientes y se estiman las medias y las varianzas. En el segundo y en los siguientes ciclos, la segmentación uniforme es reemplazada por un alineamiento Viterbi. Los parámetros iniciales procesados por HInit se vuelven a re-estimar a través de HRest. Otra vez, se utilizan los datos de referencia pero esta vez el procedimiento "segmental k-means" es reemplazado por un proceso Baum-Welch de re-estimación. [8]

Cuando no hay datos de referencia iniciales disponibles, todos los modelos son inicializados de forma idéntica y con la media y la varianza del estado igual que la media y la varianza global del audio. La herramienta que se encarga de este proceso es HCompV.

Una vez que se han creado los modelos iniciales, se utiliza la herramienta HERest para hacer un entrenamiento embebido utilizando el conjunto de entrenamiento completo. HERest hace una re-estimación Baum-Welch simple del conjunto completo de HMMs simultáneamente.

HERest es el núcleo de la herramienta de entrenamiento de HTK y está diseñado para procesar grandes bases de datos.

La herramienta HHed modifica grupos de HMMs aplicando diferentes parámetros e incrementando el número de componentes mezclados (Gaussianas) en distribuciones específicas para que HERest vuelva a re-estimar los parámetros de dichos grupos. [8]

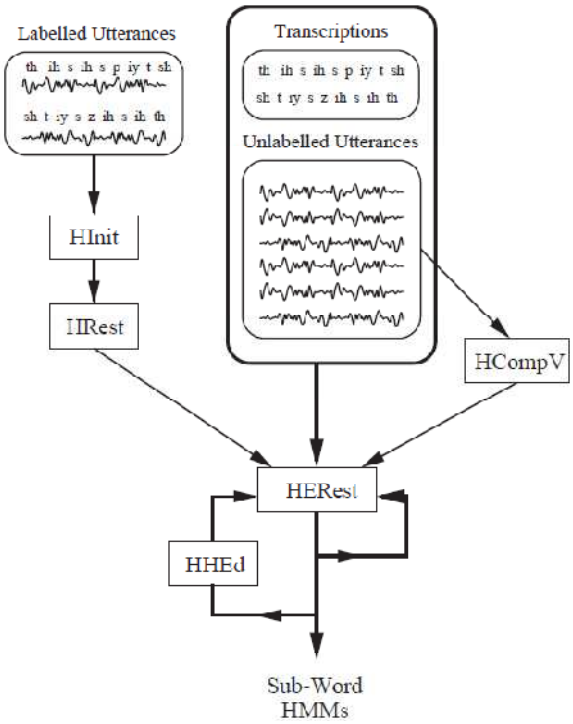


Figura 9. Herramientas de entrenamiento. Construcción de HMM

2.2.3.3 Reconocedor

HTK dispone de una herramienta de reconocimiento llamada HVite basada en el algoritmo de Viterbi. HVite recibe como input una red de secuencias de palabras permitidas, un diccionario que define cómo se pronuncia cada palabra y un conjunto de HMMs.

Funciona convirtiendo la red de palabras en una red de fonemas añadiendo la apropiada definición del HMM a cada instancia de fonema.

El reconocimiento puede hacerse sobre una lista de archivos de audio o sobre una entrada de audio directa.

Las redes de palabras que se necesitan para trabajar con HVite son normalmente simples loops de palabras en los cuales cualquier palabra puede seguir a las otras o son grafos que representan una gramática de estados finitos. Estas redes se almacenan usando el formato estándar de lattices HTK (formato de texto, editable con un simple editor de textos)

Como base para el reconocimiento se utiliza el algoritmo de Viterbi. Este algoritmo trata de encontrar el mejor camino a través de una matriz donde el eje vertical representa los estados del HMM y el eje horizontal representa los frames del audio (tiempo).

El objetivo es determinar la secuencia óptima (de mayor probabilidad) de estados dada la observación acústica y el modelo oculto de Markov. Es decir, se busca la alineación de la observación con el modelo, asignando cada vector a un estado del modelo. [8]

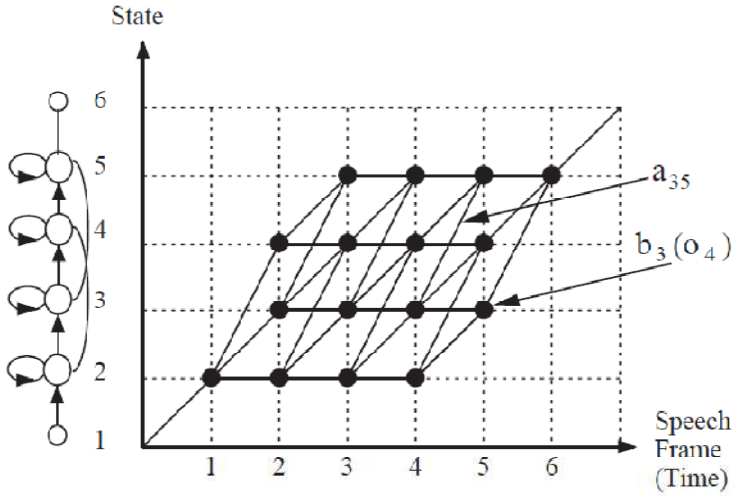


Figura 10. Algoritmo de Viterbi para la búsqueda del camino con mayor probabilidad. [9]

2.3 Detectores de Palabra Clave. Spoken Term Detection

2.3.1 Detectores de Palabra Clave

Los sistemas de detección de palabras clave o Wordspotting se componen de un conjunto de procedimientos que tratan de identificar palabras concretas dentro de una señal de audio.

Estos sistemas se pueden clasificar en tres tipos diferentes: los basados en reconocedores de habla continua de gran vocabulario (LVCSR), basados en modelos de relleno y basados en reconocedores de subunidades de palabra.

2.3.1.1 Sistemas de reconocimiento de gran vocabulario

Los sistemas de reconocimiento de gran vocabulario o LVCSR (Large Vocabulary Continuous Speech Recognition) tratan de conseguir un reconocimiento de habla pronunciada de manera natural, cubriendo un vocabulario extenso. Se caracterizan por disponer de un extenso vocabulario e incluir funciones de extracción, transcripciones automáticas del habla, modelado del lenguaje y entendimiento del habla.

Estos sistemas funcionan muy bien en el caso de que todas las palabras a reconocer formen parte del vocabulario del sistema. Si la palabra a buscar no forma parte del vocabulario del sistema no se puede puede encontrar, por lo que la palabra se considera fuera de vocabulario (OOV – Out Of Vocabulary –).

Ventajas:

- Es el que mejor funciona en caso de que todas las palabras que forman parte de la consulta del usuario formen parte del vocabulario del sistema, ya que se puede hacer uso de un modelo de lenguaje de palabra entrenado a partir de textos muy grandes (de millones de palabras).
- Búsquedas rápidas.
- Proporciona información útil para la presentación de contenidos de audio sin necesidad de reproducirlo.

Inconvenientes:

- Imposibilidad de ser usado cuando una palabra a buscar no está en el diccionario.
- Problemas de robustez frente a variaciones del tipo de audio o del tipo de habla: Tasa de error elevada cuando las condiciones del audio y la voz sobre las que trabaja el reconocedor no coinciden con las condiciones de entrenamiento del mismo. [1]

2.3.1.2 Sistemas basados en modelos de relleno

Son sistemas donde el vocabulario consta de las palabras clave a buscar y los modelos de relleno sólo intentan absorber las palabras del audio que no son clave.

El proceso de decodificación acústica propone la secuencia más probable de palabras existentes en el audio. En este caso, hay que tener en cuenta las palabras clave y cualquier otro tipo de sonido, palabra o efecto que pueda aparecer en el audio. Por este motivo, los modelos de relleno son utilizados para llenar los intervalos de habla con ausencia de palabras clave (el entrenamiento de los modelos de relleno es fundamental si se quiere conseguir un mejor rendimiento del sistema. Para ello, se han investigado modelos de relleno basados en fonemas, grafemas, sílabas, clases amplias, palabras, etc).

La salida de estos sistemas está compuesta por el conjunto de palabras clave propuestas durante el proceso de decodificación junto con los modelos de relleno que dicho proceso propone cuando considera que en cierta región de voz no existe ninguna palabra clave.

Aunque el proceso de decodificación acústica de estos sistemas obtiene un rendimiento aceptable, se han investigado medidas de confianza adicionales para tratar de eliminar aquellas hipótesis (palabras clave) propuestas por dicho proceso que no tuviesen una puntuación (score) deseada [1]. A continuación presentamos varias de estas medidas de confianza:

- Medida de confianza a partir de la puntuación dada a la palabra clave durante el proceso de decodificación (de tal forma que las hipótesis que no tengan una puntuación por encima de un determinado umbral son rechazadas de la salida final del sistema).[19]

- Uso de redes neuronales a partir de ciertas características de entrada obtenidas durante el proceso de decodificación (puntuación global de la palabra, puntuación de las N-mejores hipótesis, etc) de tal forma que aquellas hipótesis que la red neuronal clasifica como falsas alarmas son rechazadas de la salida final del sistema.[20]

- Medida de confianza basada en la semejanza que existe entre la salida de un reconocedor de fonemas en aquellas regiones de la señal de voz donde el reconocedor de palabras clave basado en modelos de relleno proponía cada hipótesis y la secuencia real de fonemas de la hipótesis dada por este último. [21]

Las palabras clave junto con los modelos de relleno entran en el módulo donde se fija una medida de confianza que se utilizará para detectar errores de reconocimiento, conceptos semánticos incorrectos y palabras fuera del vocabulario. Con las medidas de confianza se consigue aumentar el rendimiento del sistema (en términos de aciertos y falsas alarmas).

Con este procedimiento, sólo se intenta reconocer unas palabras determinadas, el resto de audio se asigna a modelos de relleno.

Ventajas:

- Es una técnica muy precisa y menos costosa que la anterior.
- Eficiente para aplicaciones cuyo vocabulario no suele sufrir cambios (operaciones bancarias, call-centers, gestión de reservas, etc).

Inconvenientes:

- Si cambia alguna palabra a buscar es necesario reprocesar todo el audio, por lo que no son eficaces para aplicaciones donde el vocabulario cambia frecuentemente.

2.3.1.3 Sistemas basados en reconocedores de sub-unidades de palabra

Los sistemas basados en reconocedores de subunidades de palabra pretenden solucionar el problema que presentan las anteriores técnicas de detección de palabras clave, principalmente el problema de las detecciones de términos que estén fuera del vocabulario del sistema. Por este motivo se desarrollaron los sistemas "Spoken Term Detection (STD)", basados en este tipo de sistemas, en los cuales nos hemos basado para este proyecto y que explicaremos más en detalle en la siguiente sección.

Estos sistemas también pueden hacer uso de procesos de medida de la confianza como las listas n-best (utilizando modelos acústicos y modelos de lenguaje para producir una lista con las n secuencias de palabras más probables en un tiempo razonable, para posteriormente re-estimar estas n hipótesis usando modelos más precisos y obtener la secuencia de palabras más probable) [22], mínima distancia de edición (número mínimo de operaciones –inserción, eliminación o sustitución– para convertir una palabra o secuencia de palabras en otra) [23] o confianza discriminativa (realizando procesamientos a través de árboles de decisión) [24].

Ventajas:

- No necesitan reprocesar el audio cuando el vocabulario de la aplicación cambia.
- Mayor velocidad de procesamiento del audio.

Inconvenientes:

- Es el sistema con peores resultados a nivel de aciertos y falsas alarmas (pero se ve compensado por la mayor velocidad de procesamiento del audio, la mayor flexibilidad para manejar vocabularios que cambian con el tiempo y la detección de palabras fuera de vocabulario).

En la actualidad los sistemas de STD suelen ser una combinación de sistemas de reconocimiento de gran vocabulario y estos últimos para poder procesar palabras tanto dentro como fuera de vocabulario.

2.3.2 Spoken Term Detection

2.3.2.1 Introducción

La arquitectura estándar de un sistema de STD (Spoken Term Detection/ Búsqueda de palabras clave en documentos hablados) consiste en un Subsistema de Reconocimiento de voz automático (ASR/RAV) que genera un lattice de palabra o fonema y un Subsistema STD que finalmente toma la decisión final sobre los términos que ha detectado.

La arquitectura estándar de un STD es: El Reconocedor de Voz (Speech Recognition) genera los lattices a partir de la señal de voz; el Detector de Términos (Term Detector) busca en esos lattices posibles ocurrencias de términos; Decisor (Decision Maker) estima si una ocurrencia es fiable.

La herramienta de NIST (National Institute of Standards and Technology) se utiliza para la evaluación de las detecciones a través de ATWV (Average Term Weighted Value) y curvas DET (Detection Error Tradeoff).

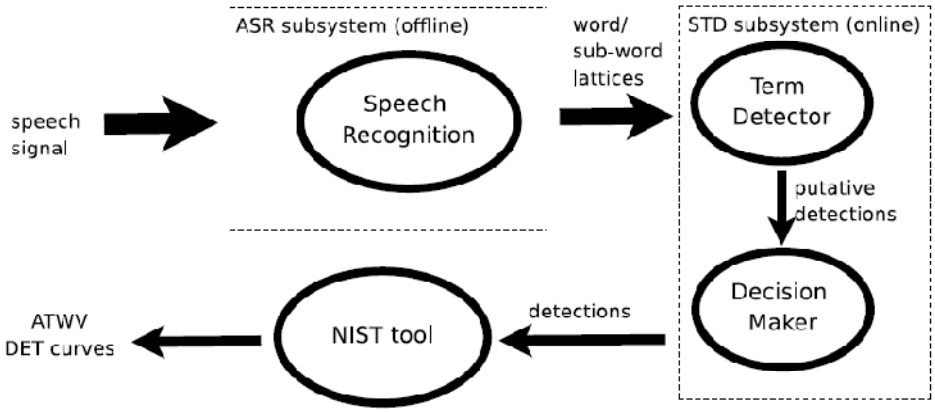


Figura 11. Arquitectura STD estándar

2.3.2.2 Detección y medida de la confianza

A partir de la búsqueda en el lattice, se consigue detectar un conjunto de términos. Se define una detección *d* como una tupla o secuencia ordenada de objetos que contiene toda la información sobre esa detección [9], ofreciendo

$$d = (K, s, v_a, v_l, \ldots) \tag{9}$$

donde K es el término detectado, v_a y v_l son los scores acústicos y de modelo de lenguaje respectivamente y s representa el intervalo de tiempo que ocupa el término detectado (y que vuelve a ser una tupla formada por el tiempo inicial y el tiempo final)

$$s = (t_{start}, t_{end}) \tag{10}$$

Hay que destacar que puede haber más información para una detección, como por ejemplo la probabilidad de pronunciación, y que se indica con "…" en la ecuación 9.

Con las detecciones posibles obtenidas, tenemos que valorar si éstas son fiables o no, basándonos en puntuaciones que midan la confianza. En la práctica la confianza basada en el lattice se calcula como:

$$c_{lat} = \frac{\sum_{\pi_\alpha,\pi_\beta} p(O|\pi_\alpha, K_{t1}^{t2}, \pi_\beta)P(\pi_\alpha, K_{t1}^{t2}, \pi_\beta)}{\sum_{\varsigma} p(O|\varsigma)P(\varsigma)} \tag{11}$$

donde π_α y π_β son caminos antes y después del término K, con π_α empezando desde el comienzo del audio y π_β terminando al final del audio. *O* representa la observación, K_{t1}^{t2} se corresponde con el camino que incluye el término K entre los instantes t_1 y t_2, y ς en el denominador representa cualquier camino dentro del lattice. [14]

2.3.2.3 Decisor (Decision Maker)

Una vez obtenida la medida de la confianza, puede determinarse si una detección posible pasa a ser una detección fiable (hit) o una falsa alarma. Este proceso lo lleva a cabo el decisor, dentro de la arquitectura STD.

Intuitivamente, el objetivo del proceso de decisión es reducir las falsas alarmas con la menor pérdida o rechazo de términos posible. Dependiendo de la aplicación tendremos mayor o menor tolerancia a las falsas alarmas, por tanto, se debe definir un umbral θ de FA/hit en el decisor.

Una aproximación simple para la decisión es comparar directamente la confianza de una detección con el umbral θ, y determinar si es fiable si y sólo si es igual o superior a θ. Esta estrategia simple se puede formular como una función de decisión segmentada:

$$\text{assert (d)} = \begin{cases} 1 & if\ c(d) \geq \theta \\ 0 & if\ c(d) < \theta \end{cases} \qquad (12)$$

donde d es la detección y c(d) es su confianza.

Una propiedad interesante de esta aproximación es que el umbral θ es independiente del término, por lo que es la estrategia de decisión en sí mismo. [9]

En este proyecto añadiremos más variables a la estrategia de decisión incluyendo el análisis de características del lattice, fonéticas, prosódicas, de duración, etc para intentar mejorar y hacer más preciso ese proceso de decisión.

2.3.2.4 Evaluación

Hay varias métricas que se usan comúnmente para evaluar el rendimiento de un sistema STD y cada una refleja aspectos diferentes del rendimiento del sistema.

- **ATWV (Average term-weighted value)**

  Se define como una media de la suma ponderada de probabilidades de rechazo y probabilidades de falsa alarma, $P_{miss}(K)$ y $P_{FA}(K)$, sobre los términos K:

$$ATWV = 1 - \frac{\Sigma_{K\in\Delta}[P_{miss}(K)+\beta P_{FA}(K)]}{|\Delta|} \qquad (13)$$

  Donde $\beta = \frac{c}{v}(P_{prior}(K)^{-1} - 1)$, y $|\Delta|$ representa el tamaño del conjunto de términos de búsqueda, Δ. La herramienta de evaluación NIST proporciona una probabilidad uniforme inicial del término, $P_{prior}(K) = 10^{-4}$, y la relación $\frac{c}{v} = 0.1$

- **Detection Error Tradeoff (DET) curve**

  Para tener una visión del rendimiento del sistema STD, las curvas DET representan gráficamente la probabilidad de rechazo frente a la probabilidad de falsa alarma. El ATWV se mide en un punto particular de la curva DET, el cual corresponde a un umbral de confianza que suele estar especificado optimizando el resultado del ATWV. Al máximo valor de ATWV sobre la curva DET se le define como *max*-ATWV y refleja el mejor rendimiento del sistema STD dentro de un umbral de confianza ideal.

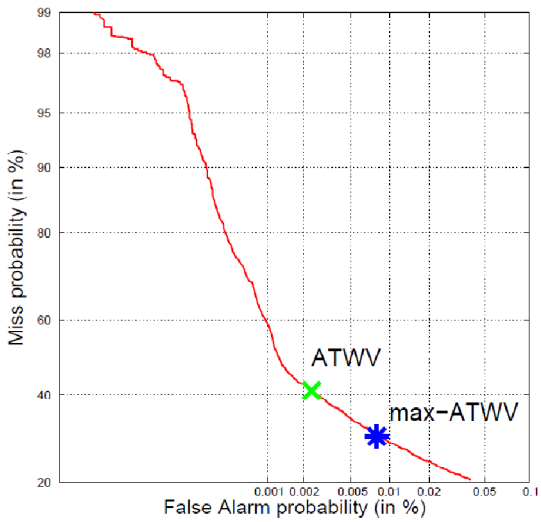


Figura 12. Ejemplo de curva DET con los valores de ATWV y max-ATWV

3

MEDIOS DISPONIBLES Y DISEÑO

3.1 Medios disponibles

3.1.1 Base de datos ALBAYZIN (Español, corpus geográfico)

La base de datos ALBAYZIN consiste en varios corpora de términos en español que han sido diseñados con el objetivo de contribuir al desarrollo y la evaluación de sistemas de reconocimiento y procesado del habla.

Para nuestros experimentos utilizamos el corpus geográfico, formado por frases correspondientes a una tarea de consulta a una base de datos geográfica. Las frases son sometidas a una fuerte restricción semántica, con el fin de incluir información relativa a este aspecto en el reconocimiento o comprensión del habla. Las construcciones sintácticas reflejan la forma natural del habla en lengua castellana. [5]

	Corpus Fonético (transcrito ortográficamente y etiquetado fonéticamente)	Corpus Geográfico (transcrito ortográficamente)
Conjunto de TRAIN	**Nombre:** Conjunto de entrenamiento fonético. **Contenido:** 4800 frases fonéticamente balanceadas de 164 hablantes: 3 horas y 20 minutos.	**Nombre:** Conjunto de desarrollo STD **Contenido:** 4400 frases de 88 hablantes: 3 horas y 40 minutos
Conjunto de TEST	**Nombre:** Conjunto de test fonético. **Contenido:** 2000 frases fonéticamente balanceadas de 40 hablantes: 1 hora y 40 minutos.	**Nombre:** Conjunto de test STD **Contenido:** 2400 frases de 48 hablantes: 2 horas

Tabla 1. Especificaciones de los corpora de la Base de Datos ALBAYZIN [5]

Disponemos de un conjunto de 4400 lattices (TRAIN) a partir del cual buscaremos términos utilizando un diccionario de 105 palabras y otro diccionario de 500 palabras. También disponemos de un conjunto de 2400 lattices (TEST), con el que utilizaremos un diccionario de 400 palabras.

3.1.2 Base de datos AMI (Inglés)

AMI es una base de datos formada por un corpus de 100 horas de voz conversacional, dividido en conversaciones entre grupos de cuatro personas. Las conversaciones son en inglés, pero una gran proporción de los hablantes no son hablantes ingleses nativos, lo que provee un alto grado de variabilidad en los patrones de voz. [6]

Disponemos de un conjunto de 1758 lattices (TRAIN), a partir del cual buscaremos términos utilizando un diccionario de 490 palabras y otro diccionario de 67 palabras. También disponemos de otro conjunto de 11092 lattices (TEST), con el que utilizaremos un diccionario de 268 palabras y otro diccionario de 484 palabras.

3.1.3 Archivos de Pitch y Energía de ficheros de audio (PRAAT)

Praat (del holandés "hablar") es un software gratuito para el análisis científico del habla usado en lingüística. Fue diseñado y continúa siendo desarrollado por Paul Boersma y David Weenink de la Universidad de Ámsterdam. Puede ser instalado en varios sistemas operativos, incluyendo Unix, Mac y Microsoft Windows (95, 98, NT4, ME, 2000, XP, Vista).

Praat es capaz de grabar la voz en varios tipos de archivos de audio y mostrar los espectrogramas. Además, permite el análisis de la entonación, la intensidad o volumen, los formantes, cocleagrama, etc. Praat también puede ser automatizado para análisis más complejos, lo que ha resultado útil para investigadores de alto nivel. Se pueden calcular valores de jitter, shimmer, entre otros, y utilizarlos para el análisis acústico.

En este proyecto se ha utilizado el software Praat para extraer la energía y el pitch de los ficheros de voz analizados. [7]

3.1.4 Hardware

- Ordenador personal Intel Core 2 CPU
  6300 @ 1.86GHz.
  (Trabajando sobre Ubuntu versión 8.04)

- NetBook AMD Athlon, L110 processor
  (Trabajando sobre Ubuntu versión 9.10 –karmic-)

3.1.5 Software

3.1.5.1 HTK (Hidden Markov Model Toolkit)

Como se ha descrito con anterioridad, HTK es un toolkit para la construcción de HMMs (Modelos Ocultos de Markov). HTK se usa principalmente para desarrollar herramientas de procesamiento de voz, en particular reconocedores de voz (aunque puede ser utilizado para cualquier otra herramienta de reconocimiento de patrones, síntesis de voz o secuencias de ADN). [8]

HTK está compuesto por un conjunto de librerías y herramientas desarrolladas en C. Las herramientas facilitan el análisis de la voz, el entrenamiento de los HMM, el test y la extracción de resultados. El software soporta la creación de HMM con distribuciones continuas de mezclas de Gaussianas o por medio de distribuciones discretas pudiendo crear de esta forma complejos sistemas de HMM. [8]

HTK fue desarrollado originalmente por el departamento de Ingeniería de la Universidad de Cambridge, conocido como el grupo “the Speech Vision and Robotics”. Contiene diferentes algoritmos de estimación de parámetros como el algoritmo Baum-Welch o el algoritmo Viterbi (este último utilizado para la decodificación, recibiendo una red de secuencias de palabras permitidas, un diccionario que define cómo se pronuncia cada palabra y un conjunto de HMMs). [8]

3.1.5.2 Software de búsqueda en lattices Sphinx

SPHINX [10] es otro de los sistemas de reconocimiento existentes en la actualidad. Fue desarrollado en la Universidad de Carnegie Mellon [11] y al igual que HTK, se basa en la construcción de Modelos Ocultos de Markov. Los componentes de SPHINX son el Sphinxtrain, para entrenamiento de los modelos, y el SPHINX decoder para reconocimiento.

El Sphinxtrain está compuesto por un conjunto de programas que han sido compilados para dos tipos de sistemas: Linux y alpha. Genera modelos acústicos discretos, semicontinuos o continuos (HMM con topología left to right) que pueden tener desde 1 hasta 24 gaussianas en cada estado.

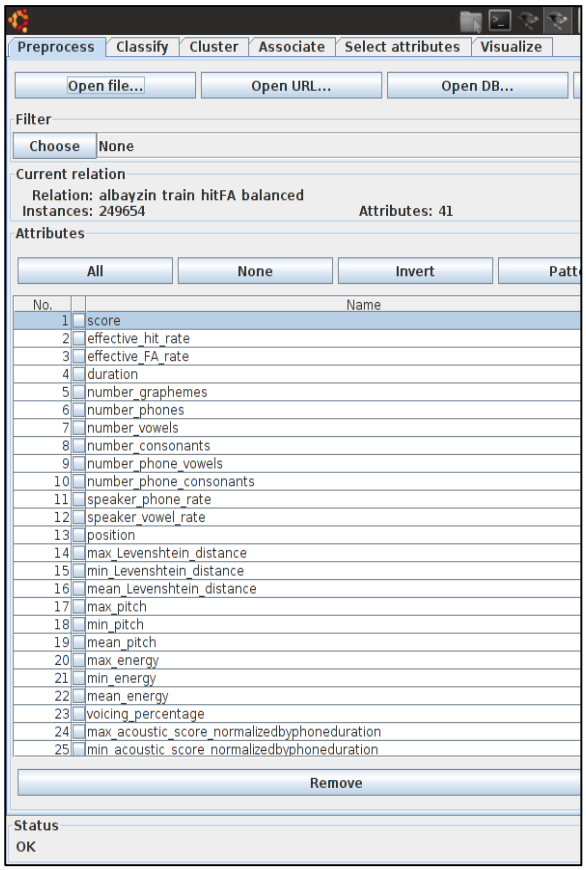
El SPHINX decoder contiene algoritmos de programación dinámica como Baum-Welch o Viterbi.

Las principales necesidades de SPHINX a la hora de realizar un reconocimiento son:

1. El diccionario de pronunciación.
2. El diccionario con sonidos de relleno.
3. Los modelos acústicos.
4. El modelo de lenguaje.
5. Los datos de prueba.

3.1.5.3 Herramienta de aprendizaje automático WEKA

Weka (Waikato Environment for Knowledge Analysis - Entorno para Análisis del Conocimiento de la Universidad de Waikato) es una plataforma de software para aprendizaje automático y minería de datos escrito en Java y desarrollado en la Universidad de Waikato. Weka es un software libre distribuido bajo licencia GNU-GPL. [12]



El paquete Weka contiene una colección de herramientas de visualización y algoritmos para análisis de datos y modelado predictivo, unidos a una interfaz gráfica de usuario para acceder fácilmente a sus funcionalidades. La versión original de Weka fue un front-end en TCL/TK para modelar algoritmos implementados en otros lenguajes de programación, más unas utilidades para preprocesamiento de datos desarrolladas en C para hacer experimentos de aprendizaje automático. Esta versión original se diseñó inicialmente como herramienta para analizar datos procedentes del dominio de la agricultura, pero la versión más reciente basada en Java (WEKA 3), que empezó a desarrollarse en 1997, se utiliza en muchas y muy diferentes áreas, en particular con finalidades docentes y de investigación.

Figura 13. Ejemplo de Preprocesamiento de características con WEKA

La interfaz "Explorer" dispone de varios paneles que dan acceso a los componentes principales del banco de trabajo [12]:

El panel "Preprocess" dispone de opciones para importar datos de una base de datos, de un fichero CSV, etc., y para preprocesar estos datos utilizando los denominados algoritmos de filtrado. Estos filtros se pueden utilizar para transformar los datos (por ejemplo convirtiendo datos numéricos en valores discretos) y para eliminar registros o atributos según ciertos criterios previamente especificados.

El panel "Classify" permite al usuario aplicar algoritmos de clasificación estadística y análisis de regresión (denominados todos clasificadores en Weka) a los conjuntos de datos resultantes, para estimar la exactitud del modelo predictivo resultante, y para visualizar predicciones erróneas, curvas ROC, etc., o el propio modelo (si este es susceptible de ser visualizado, como por ejemplo un árbol de decisión).

El panel "Cluster" da acceso a las técnicas de clustering o agrupamiento de Weka como por ejemplo el algoritmo k-means. Éste es sólo una implementación del algoritmo expectación-maximización para aprender una mezcla de distribuciones normales.

El panel "Associate" proporciona acceso a las reglas de asociación aprendidas que intentan identificar todas las interrelaciones importantes entre los atributos de los datos.

El panel "Select attributes" proporciona algoritmos para identificar los atributos más predictivos en un conjunto de datos.

El panel "Visualize" muestra una matriz de puntos dispersos (Scatterplot) donde cada punto individual puede seleccionarse y agrandarse para ser analizado en detalle usando varios operadores de selección. [12]

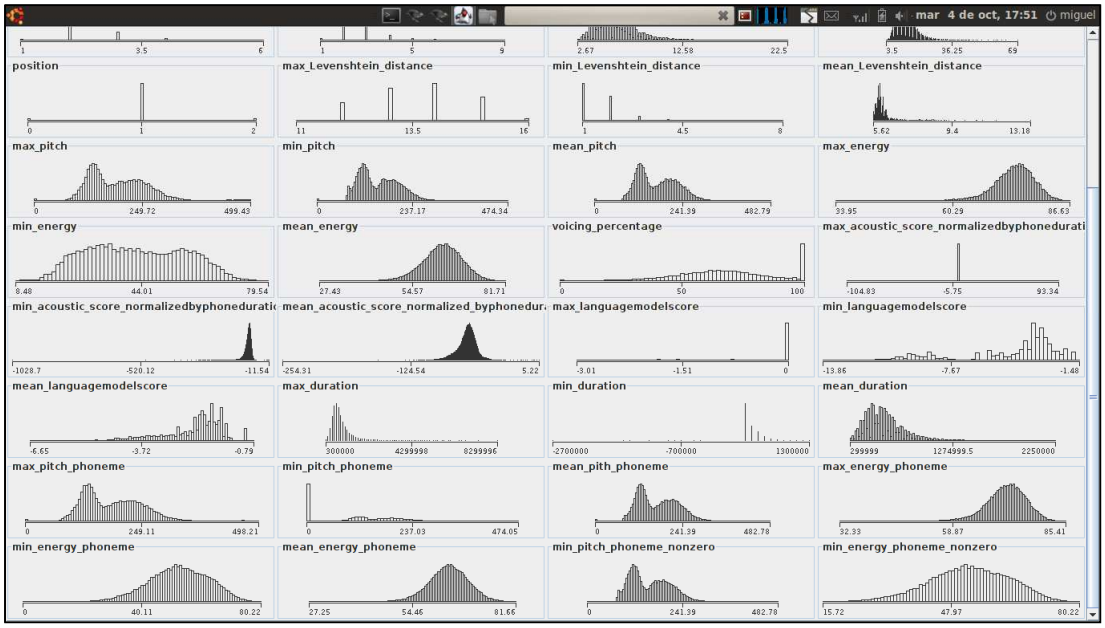


Figura 14. Ejemplo de visualización de características en WEKA

3.2 Diseño

En este trabajo, y dado un sistema STD (Spoken Term Detection) comprendido por un Subsistema ASR (Automatic Speech Recognizer) y un Subsistema STD (Detección de términos), nos hemos centrado en la parte de búsqueda dentro de los lattices generados por el ASR, en la extracción de características fonéticas basadas en información de los lattices (duraciones, puntuaciones acústicas y de modelo de lenguaje), en la extracción de características fonéticas (pitch y energía) combinando información de los lattices con información de documentos generados por la herramienta PRAAT [7] (de pitch y energía por frame) y en el análisis de las características más determinantes para el Decisor, a la hora de determinar si realmente ha detectado un término (a través de Análisis de la Varianza explicada por cada característica, y Evaluación y Clasificación de Atributos con la herramienta WEKA de Aprendizaje Automático).

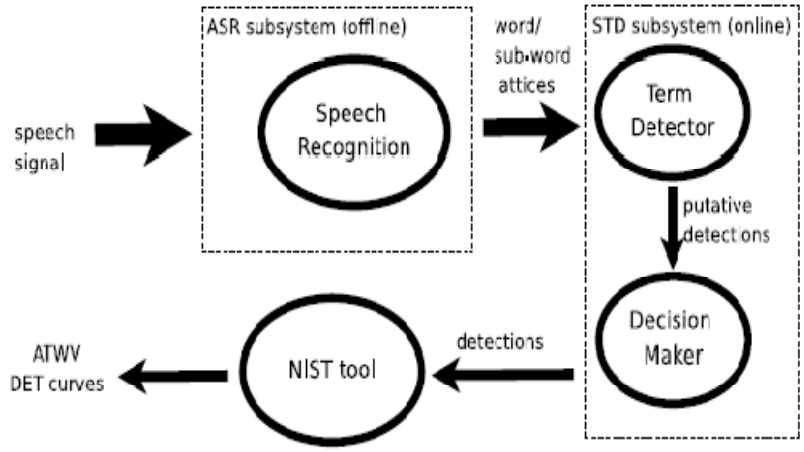


Figura 15. Arquitectura de un STD estándar

3.2.1 Detección de palabras/fonemas en lattices

Adaptamos la herramienta de búsqueda en lattices de Sphinx proporcionada por el grupo de investigación ATVS a nuestro formato de lattices HTK. El objetivo es obtener un fichero que muestra las detecciones de las transcripciones fonéticas de las palabras incluidas en el fichero de vocabulario que le proporcionamos y otro fichero similar pero aislando la información de cada fonema dentro de la palabra.

3.2.2 Extracción de características iniciales

Las características de las que partimos como referencia, y que han sido ya evaluadas en pasados trabajos [13], son las siguientes:

- Características basadas en información del lattice (LAT): Este conjunto de características comprende la confianza para cada detección, R0 (probabilidad de acierto), R1 (probabilidad de falsa alarma)

$$R_0(K) = \frac{\sum_i c_f(d_i^K)}{T} \quad (14) \qquad R_1(K) = \frac{\sum_i (1 - c_f(d_i^K))}{T} \quad (15)$$

$c_f(d_i^K)$ representa la confianza basada en el lattice de la detección i del término K y T es la duración total del audio. [14]

En la práctica la confianza basada en el lattice se calcula como:

$$c_{lat} = \frac{\sum_{\pi_\alpha,\pi_\beta} p(O|\pi_\alpha, K_{t1}^{t2}, \pi_\beta) P(\pi_\alpha, K_{t1}^{t2}, \pi_\beta)}{\sum_\varsigma p(O|\varsigma) P(\varsigma)} \quad (16)$$

donde π_α y π_β son caminos antes y después del término K, con π_α empezando desde el comienzo del audio y π_β terminando al final del audio. O representa la observación, K_{t1}^{t2} se corresponde con el camino que incluye el término K entre los instantes t_1 y t_2, y ς en el denominador representa cualquier camino dentro del lattice. [14]

1	score
2	effective_hit_rate
3	effective_FA_rate
4	duration
5	number_graphemes
6	number_phones
7	number_vowels
8	number_consonants
9	number_phone_vowels
10	number_phone_consonants
11	speaker_phone_rate
12	speaker_vowel_rate
13	position
14	max_Levenshtein_distance
15	min_Levenshtein_distance
16	mean_Levenshtein_distance
17	max_pitch
18	min_pitch
19	mean_pitch
20	max_energy
21	min_energy
22	mean_energy
23	voicing_percentage

Figura 16. Lista de características. [4]

- Características léxicas (LEX): Este conjunto de características está comprendido por el número total de grafemas, los grafemas vocales, grafemas consonantes, fonemas, fonemas vocales y fonemas consonantes de cada término.

- Características de la Distancia de Levenshtein (LEV): Distancia de Levenshtein mínima, máxima y media para cada término con respecto al resto.

- Características de Duración (DUR): Duración de cada detección, duración dividida entre el número de fonemas (phone speech rate) y dividida entre el número de vocales (vowel speech rate) de cada detección.

- Posición (POS): Representa si la detección fue encontrada la primera en el lattice, la última en el lattice o en otra posición.

- Prosódicas (PROS): Pitch (máximo, mínimo y medio para cada detección), Energía (máxima, mínima y media para cada detección) y el porcentaje de voz (porcentaje de voz para cada detección dentro de la señal de voz).

A partir de los resultados obtenidos en pasados estudios [13], identificamos una serie de características a nivel de fonema que podrían ser interesantes a la hora de mejorar el proceso de reconocimiento de una palabra. Además, este tipo de información (características a nivel de fonema extraídas a partir de lattices) nunca había sido aislada y extraída dentro de la búsqueda en lattices.

24	max_acoustic_score_normalizedbyphoneduration
25	min_acoustic_score_normalizedbyphoneduration
26	mean_acoustic_score_normalized_byphoneduration
27	max_languagemodelscore
28	min_languagemodelscore
29	mean_languagemodelscore
30	max_duration
31	min_duration
32	mean_duration
33	max_pitch_phoneme
34	min_pitch_phoneme
35	mean_pith_phoneme
36	max_energy_phoneme
37	min_energy_phoneme
38	mean_energy_phoneme
39	min_pitch_phoneme_nonzero
40	min_energy_phoneme_nonzero

Una vez extraídos y aislados los nodos "fonema" que conforman la palabra detectada en el lattice, seremos capaces de extraer características de cada fonema. Principalmente acabarán interesándonos en este trabajo las características de Duración y Prosódicas (éstas últimas serán el pitch y la energía).

Figura 17. Lista de nuevas características

3.2.3 Evaluación y clasificación de características

Tras extraer las nuevas características a nivel de fonema, las adjuntamos al conjunto de características utilizadas en anteriores estudios [13] (éstas a nivel de palabra) para tratar de mejorar el porcentaje de acierto del Decisor. A través de Análisis de Varianza explicada por cada característica y/o conjunto de características, Evaluaciones de Atributos y Clasificación a través de un Árbol de Decisión (utilizando la herramienta WEKA de Aprendizaje Automático) comprobaremos si éstas nuevas características podrían aportar una mejora a la hora de determinar finalmente el reconocimiento de un término.

4

DESARROLLO

4.1 Modificación de la herramienta de Búsqueda en lattice para HTK (basado en SPHYNX)

Suponiendo un reconocedor de voz ya listo y entrenado que devuelve unos lattices fonéticos o de palabra, el objetivo es encontrar y/o ajustar parámetros que permitan determinar si una hipótesis de ocurrencia de una palabra es un acierto o un error, de la manera más precisa posible, a través de la información contenida en dichos lattices u otra información adicional.

4.1.1 Formato Lattice Sphynx

```
#
Frames 280
#
Nodes 806 (NODEID WORD STARTFRAME FIRST-ENDFRAME LAST-ENDFRAME ASCORE LSCORE)
0 </s> 280 280 280
1 d 274 276 277
2 m 272 274 275
3 x 272 274 275
4 !r 272 274 275
5 m 271 273 275
6 b 271 273 275
7 y 271 273 275
8 r 271 273 275
9 d 271 273 275
10 x 271 273 275
11 n 271 273 275
12 y 268 270 273
13 g 268 270 271
14 <sil> 267 274 276
15 C 267 269 270
```

```
#
Edges (FROM-NODEID TO-NODEID ASCORE)
5 1 -100999
6 1 -114839
7 1 -108455
8 1 -87633
9 1 -85590
10 1 -99566
11 1 -89304
12 1 -158491
12 5 -104825
12 6 -104825
12 7 -104825
12 8 -104825
12 9 -104825
12 10 -104825
12 11 -104825
13 2 -111476
```

La estructura de un lattice generado a través de un reconocedor de voz SPHYNX se podría dividir en dos bloques: en el primero, cada registro presenta el número de nodo, el fonema reconocido, el frame inicial y el primer y último frame final (el fonema realmente habrá acabado en un frame entre estos dos últimos).
En el segundo bloque se muestra información del Arco entre Nodos (nodo inicial, nodo final y Puntuación acústica asociada al Arco).

Figura 18. Ejemplo de Lattice Sphynx

Frames (10^{-2} segundos)
AScore (puntuación acústica)
LScore (Puntuación basada en modelo de lenguaje)

4.1.2 Formato Lattice HTK

```
N=5037 L=24018
I=0     t=0.00  W=!NULL
I=1     t=0.02  W=R1                    v=1
I=2     t=0.02  W=R1                    v=1
I=3     t=0.02  W=R1                    v=1
I=4     t=0.02  W=R1                    v=1
I=5     t=0.02  W=R1                    v=1
I=6     t=0.02  W=R1                    v=1
I=7     t=0.02  W=R1                    v=1
I=8     t=0.02  W=R1                    v=1
I=9     t=0.02  W=R1                    v=1
I=10    t=0.02  W=R1                    v=1
I=11    t=0.02  W=R1                    v=1
I=12    t=0.03  W=_                     v=1
I=13    t=0.03  W=_                     v=1
I=14    t=0.03  W=_                     v=1
I=15    t=0.03  W=_                     v=1
I=16    t=0.03  W=
```

La estructura de un lattice generado a través de un reconocedor de voz de HTK se podría dividir en dos bloques: en el primero, cada registro presenta el número de Nodo (I), el tiempo de inicio dentro del archivo de audio (t -en segundos-), y el fonema/silencio que se ha reconocido (W). "R1" y "_" representan silencio largo y silencio corto respectivamente.

Cada registro del segundo bloque del lattice muestra el identificador del arco entre nodos, el nodo de inicio, nodo de fin, puntuación acústica y puntuación del modelo de lenguaje.

```
I=5025 t=2.15  W=e                     v=1
I=5026 t=2.15  W=a                     v=1
I=5027 t=2.16  W=a                     v=1
I=5028 t=2.16  W=D                     v=1
I=5029 t=2.16  W=T                     v=1
I=5030 t=2.16  W=n                     v=1
I=5031 t=2.16  W=e                     v=1
I=5032 t=2.16  W=t                     v=1
I=5033 t=2.16  W=_                     v=1
I=5034 t=2.21  W=s                     v=1
I=5035 t=2.36  W=!NULL
I=5036 t=2.36  W=R2                    v=1
J=0     S=0     E=1     a=-91.78     l=0.000
J=1     S=0     E=2     a=-91.78     l=0.000
J=2     S=0     E=3     a=-91.78     l=0.000
J=3     S=0     E=4     a=-91.78     l=0.000
J=4     S=0     E=5     a=-91.78     l=0.000
J=5     S=0     E=6     a=-91.78     l=0.000
J=6     S=0     E=7     a=-91.78     l=0.000
J=7     S=0     E=8     a=-91.78     l=0.000
J=8     S=0     E=9     a=-91.78     l=0.000
J=9     S=0     E=10    a=-91.78     l=0.000
J=10    S=0     E=11    a=-91.78     l=0.000
J=11    S=1     E=12    a=-56.99     l=-6.920
J=12    S=1     E=13    a=-56.99     l=-6.920
J=13    S=1     E=14    a=-56.99     l=-6.920
J=14    S=1     E=15    a=-56.99     l=-6.920
J=15    S=1     E=16    a=-56.99     l=-6.920
J=16    S=1     E=17    a=-56.99     l=-6.920
J=17    S=1     E=18    a=-56.99     l=-6.920
```

Figura 19. Ejemplo de Lattice HTK

4.1.3 Búsqueda en Lattice (HTK)

Desarrollamos un programa (en C) que es capaz de realizar una búsqueda de términos a partir de una lista de lattices fonéticos en formato HTK (basado en otro programa ya desarrollado de búsqueda en lattices Sphynx) y que es capaz de aislar los fonemas para poder extraer posteriormente la información a nivel de fonema (esta información corresponde a las características a nivel de fonema extraídas del lattice que se detallarán en la siguiente sección junto con el resto de nuevas características).

4.1.3.1 Input/Output

Ejecutamos el programa principal pasándole como argumentos el word_map (INPUT), archivo con la lista de lattices (INPUT), el vocabulario (INPUT) y la lista de silencios (INPUT), archivo.mlf de detección de fonemas (OUTPUT), archivo.mlf de detección de palabras (OUTPUT) y archivo de características fonéticas (OUTPUT).

INPUTs

```
Name=Spanish
SeqNo=13
Entries=51
Fields=ID
Language=Spanish
EscMode=RAW
\Words\
a       65536
e       65537
i       65538
o       65539
u       65540
A       65541
E       65542
I       65543
O       65544
U       65545
an      65546
en      65547
in      65548
on      65549
un      65550
An      65551
En      65552
In      65553
On      65554
Un      65555
b       65556
B       65557
T/      65558
d       65559
D       65560
f       65561
g       65562
G       65563
X       65564
j       65565
J/      65566
J       65567
k       65568
l       65569
L       65570
m       65571
n       65572
N       65573
Nn      65574
p       65575
r       65576
R       65577
s       65578
t       65579
T       65580
w       65581
gs      65582
R1      65583
R2      65584
_       65585
!NULL   65586
```

Figura 23. Wordmap

```
/home/miguel/Desktop/Albayzin/pfc/EAGE0301.lat
/home/miguel/Desktop/Albayzin/pfc/EAGE0349.lat
/home/miguel/Desktop/Albayzin/pfc/EAGE0397.lat
/home/miguel/Desktop/Albayzin/pfc/EAGE0445.lat
```

Figura 20. Extracto de Lista de lattices

```
alturas a l t U r a s
comunidad       k o m un n i D A D
dime    d I m e
rios    R I o s
nombre  n On m b r e
comunidades     k o m un n i D A D e s
pasa    p A s a
autonoma        a u t O n on m a
ciudades        T i u D A D e s
sistema s i s t E m a
```

Figura 21. Extracto de archivo de Vocabulario (ALBAYZIN)

```
R1
R2
_
```

Figura 22. Archivo de Silencios (ALBAYZIN)

OUTPUTs

```
#!MLF!#
"*//home/miguel/Desktop/lattices_material/latticesTRAIN/VEGE2282.lat"
27500000 28200000 k cual 0.000000 0.000000
28200000 28500000 w -297.649994 -2.820000
28500000 29100000 a -459.190002 -1.610000
29100000 30200000 l -1020.080017 -3.310000
6100000 6800000 T cien 0.000000 0.000000
6800000 10100000 j -404.019989 -1.400000
10100000 10599999 E -394.320007 -0.790000
10599999 11799999 n -992.869995 -1.790000
23599998 24200000 A alto 0.000000 0.000000
24200000 24600000 l -399.779999 -3.070000
24600000 25000000 t -420.350006 -3.620000
25000000 25700000 o -252.919998 -9.540000
.
"*//home/miguel/Desktop/lattices_material/latticesTRAIN/EBGE2318.lat"
10599999 10900000 k cual 0.000000 0.000000
10900000 11400000 w -380.149994 -2.820000
11400000 11900001 a -349.549988 -1.610000
11900001 13000000 l -898.780029 -3.310000
11400000 12300000 R rias -315.589996 -5.510000
12300000 13000000 I -499.480011 -2.230000
13000000 13500000 a -363.540009 -1.590000
13500000 13800000 s -258.890015 -2.330000
5200000 5800000 t todos 0.000000 0.000000
5800000 6400000 O -494.010010 -2.940000
6400000 7100000 D -637.929993 -2.520000
7100000 8000000 o -706.650024 -1.460000
8000000 8800000 s -664.919983 -1.290000
10300000 10900000 g guadiana 0.000000 0.000000
10900000 11400000 w -372.579987 -1.790000
11400000 11900001 a -357.769989 -1.610000
11900001 12400000 D -376.739990 -4.210000
12400000 13000000 j -405.049988 -3.240000
13000000 13600000 A -402.649994 -2.550000
13600000 14100000 n -380.399994 -1.900000
14100000 15400000 a -1036.099976 -2.600000
7000000 7400000 a arosa 0.000000 0.000000
7400000 7700000 r -246.550003 -3.240000
7700000 8000000 o -257.760010 -4.580000
8000000 8300000 s -263.779999 -2.520000
8300000 8700000 a -334.350006 -3.200000
.
```

```
#!MLF!#
"*//VEGE2282.rec"
27500000 30200000 cual
6100000 11799999 cien
23599998 25700000 alto
.
"*//EBGE2318.rec"
10599999 13000000 cual
11400000 13800000 rias
5200000 8800000 todos
10300000 15400000 guadiana
7000000 8700000 arosa
.
```

Figura 24. Archivo .mlf a nivel de palabra

Figura 25. Archivo .mlf a nivel de fonema (nunca extraído anteriormente)

```
#Extracted features based on phonemes information of each word
#word file max_ascore_normalizedbyphoneduration min_ascore_normalizedbyphoneduration ascore_normalizedbyphoneduration_average max_lmscore min_lmscore lmscore
cual /home/miguel/Desktop/lattices_material/latticesTRAIN/VEGE2282. 0.0000 -99.2167 -67.1207 0.0000 -3.3100 -1.9350 1100000 300000 675000
cien /home/miguel/Desktop/lattices_material/latticesTRAIN/VEGE2282. 0.0000 -82.7392 -43.4616 0.0000 -1.7900 -0.9950 3300000 499999 1424999
alto /home/miguel/Desktop/lattices_material/latticesTRAIN/VEGE2282. 0.0000 -105.0875 -60.2910 0.0000 -9.5400 -4.0575 700000 400000 525000
cual /home/miguel/Desktop/lattices_material/latticesTRAIN/EBGE2318. 0.0000 -81.7073 -56.9118 0.0000 -3.3100 -1.9350 1099999 300001 600000
rias /home/miguel/Desktop/lattices_material/latticesTRAIN/EBGE2318. -35.0656 -86.2967 -66.3561 -1.5900 -5.5100 -2.9150 900000 300000 600000
todos /home/miguel/Desktop/lattices_material/latticesTRAIN/EBGE2318. 0.0000 -91.1329 -67.0199 0.0000 -2.9400 -1.6420 900000 600000 720000
guadiana /home/miguel/Desktop/lattices_material/latticesTRAIN/EBGE2318. 0.0000 -79.7000 -63.9768 0.0000 -4.2100 -2.2375 1300000 499999 637500
arosa /home/miguel/Desktop/lattices_material/latticesTRAIN/EBGE2318. 0.0000 -87.9267 -67.9235 0.0000 -4.5800 -2.7080 400000 300000 340000
```

Figura 26. Archivo de características a nivel de fonema

4.1.3.2 Pseudocódigo

//Como indicamos anteriormente, ejecutamos el programa pasándole como argumentos el word_map (INPUT), archivo con la lista de lattices (INPUT), el vocabulario (INPUT) y la lista de silencios (INPUT), archivo.mlf de detección de fonemas (OUTPUT), archivo.mlf de detección de palabras (OUTPUT) y archivo de características fonéticas:

```
Main.c
{
        vocabulary = read_file_to_memory  //Lee el vocabulario en memoria
        wmap = read_wmap                  //Lee el word_map en memoria
        wmapmarksilences                  //Marca los silencios en el word_map
        while (lista_lattices)            //Una iteración por cada Lattice de la lista
        {
           read_htk_lattice          //Leemos el Lattice (Nodos y Arcos)
           while (vocabulary)        //Una iteración por cada palabra del vocabulario
           {
                htk_find_trans_in_lattice //Buscamos la palabra en el lattice
                {
                        // Haremos una búsqueda recursiva a partir de todos los nodos del lattice
                        que se corresponden con el primer fonema de la palabra.
                        for (n_node != transcription[0])
                        {
                        //Búsqueda de la secuencia de fonemas que forman la palabra
                                htk_recursively_find_trans_in_lattice
                        }
                        If (num_hypothesis>0)
                        {
                                //Organizamos las hipótesis que se han obtenido de la palabra.
                                for (num_hypothesis)
                                {
                                   //Agrupamos las hipótesis que tienen el mismo start_frame,
                                     positions_same_start_frame
                                     //Seleccionamos la hipótesis más larga,
                                     best_hypo
                                     //Localizamos los scores en los arcos y se los asignamos a
                                     cada nodo, de la mejor hipótesis
                                     nodos_final
                                }
                                //Desechamos una hipótesis que se solape en el tiempo con otra
                                (sólo se rechazan hipótesis solapadas correspondientes al mismo
                                término), extraemos las características a nivel de fonema  e
                                imprimimos los ficheros de salida.
                                Htk_print_mlf_phonemes_file
                                Htk_print_mlf_words_file
                                Htk_print_features
                        }
                }
           }
        }
}
```

4.2 Extracción de Características a Nivel de Fonema

Identificamos una serie de características a nivel de fonema que podrían ser interesantes a la hora de mejorar el proceso de reconocimiento de una palabra (basados en anteriores estudios realizados utilizando características a nivel de palabra [13]). Además, este tipo de información (características a nivel de fonema extraídas a partir de lattices) nunca había sido aislada y extraída dentro de la búsqueda en lattices.

4.2.1 Características identificadas y extraídas

Las características elegidas fueron las siguientes:

max_ascore_normalizedbyphoneduration: Máximo score acústico normalizado por la duración del fonema: Lo obtenemos normalizando el score acústico de cada fonema (extraído del lattice) por la duración del fonema e identificando finalmente el máximo de este valor con respecto a todos los fonemas dentro de una palabra.

min_ascore_normalizedbyphoneduration: Mínimo score acústico normalizado por la duración del fonema: Lo obtenemos normalizando el score acústico de cada fonema (extraído del lattice) por la duración del fonema e identificando finalmente el mínimo de este valor con respecto a todos los fonemas dentro de una palabra.

ascore_normalizedbyphoneduration_average: Score acústico medio normalizado por la duración del fonema: Normalizamos el score acústico de cada fonema (extraído del lattice) por la duración del fonema, sumamos estos valores correspondientes a los fonemas que forman un palabra y dividimos entre el número de fonemas de esa palabra.

max_lmscore: Máximo score de modelo de lenguaje: Se obtiene a través de los scores de modelo de lenguaje extraídos del lattice e identificando el valor máximo dentro de los fonemas que forman una palabra.

min_lmscore: Mínimo score de modelo de lenguaje: Lo obtenemos a través de los scores de modelo de lenguaje extraídos del lattice e identificando el valor mínimo dentro de los fonemas que forman una palabra.

lmscore_average: Media del score de modelo de lenguaje: Lo obtenemos a través de los scores de modelo de lenguaje extraídos del lattice y haciendo la media dentro de los fonemas que forman una palabra (suma de scores de modelo de lenguaje de todos los fonemas que forman la palabra entre número de fonemas que forman dicha palabra).

max_duration: Duración máxima de fonema (dentro de una palabra): Se obtiene aislando el tiempo final e inicial de cada fonema (extraídos del lattice), calculando la duración de cada fonema e identificando el valor máximo con respecto a todos los fonemas dentro de una palabra.

min_duration: Duración mínima de fonema (dentro de una palabra): Se obtiene aislando el tiempo final e inicial de cada fonema (extraídos del lattice), calculando la duración de cada fonema e identificando el valor mínimo con respecto a todos los fonemas dentro de una palabra.

duration_average: Duración media de fonema (dentro de una palabra): Se obtiene aislando el tiempo final e inicial de cada fonema (extraídos del lattice), calculando la duración de cada fonema y finalmente haciendo la media respecto al número de fonemas dentro de una palabra.

max_pitch: Pitch máximo (a nivel de fonema, dentro de una palabra):

- Para cada fonema, utilizamos la información de tiempo final y tiempo inicial que hemos obtenido con nuestro programa de búsqueda en lattices HTK.

  Después contrastamos esta información con los valores de pitch extraídos a través de Praat [7]. Estos valores son a nivel de frame, no a nivel de fonema, por lo que para calcular el valor de pitch de cada fonema hacemos la media de los valores de pitch de todos los frames contenidos en el tiempo que dura el fonema. Esa media será nuestro valor de pitch para ese fonema.

  Finalmente identificamos el valor máximo de esos valores de pitch de fonema con respecto a todos los fonemas que forman la palabra y obtenemos nuestro valor "max_pitch" (todo este proceso lo hacemos a través de una herramienta desarrollada también expresamente para este proyecto y que describiremos en el siguiente punto).

min_pitch: Pitch mínimo (a nivel de fonema, dentro de una palabra):

- Para cada fonema, utilizamos la información de tiempo final y tiempo inicial que hemos obtenido con nuestro programa de búsqueda en lattices HTK.

  Después contrastamos esta información con los valores de pitch extraídos a través de Praat [7]. Estos valores son a nivel de frame, no a nivel de fonema, por lo que para calcular el valor de pitch de cada fonema hacemos la media de los valores de pitch de todos los frames contenidos en el tiempo que dura el fonema. Esa media será nuestro valor de pitch para ese fonema.

Finalmente identificamos el valor mínimo de esos valores de pitch de fonema con respecto a todos los fonemas que forman la palabra y obtenemos nuestro valor "min_pitch".

mean_pitch: Pitch medio (a nivel de fonema, dentro de una palabra):

- Para cada fonema, utilizamos la información de tiempo final y tiempo inicial que hemos obtenido con nuestro programa de búsqueda en lattices HTK.

  Después contrastamos esta información con los valores de pitch extraídos a través de Praat [7]. Estos valores son a nivel de frame, no a nivel de fonema, por lo que para calcular el valor de pitch de cada fonema hacemos la media de los valores de pitch de todos los frames contenidos en el tiempo que dura el fonema. Esa media será nuestro valor de pitch para ese fonema.

  Finalmente hacemos la media de esos valores por fonema con respecto al número de fonemas que forman la palabra y obtenemos nuestro valor "mean_pitch".

max_intensity: Energía máxima (a nivel de fonema, dentro de una palabra):

- Para cada fonema, utilizamos la información de tiempo final y tiempo inicial que hemos obtenido con nuestro programa de búsqueda en lattices HTK.

  Después contrastamos esta información con los valores de energía extraídos a través de Praat [7]. Estos valores son a nivel de frame, no a nivel de fonema, por lo que para calcular el valor de energía de cada fonema hacemos la media de los valores de energía de todos los frames contenidos en el tiempo que dura el fonema. Esa media será nuestro valor de energía para ese fonema.

  Finalmente identificamos el valor máximo de esos valores de energía de fonema con respecto a todos los fonemas que forman la palabra y obtenemos nuestro valor "max_intensity" (todo este proceso lo hacemos a través de una herramienta desarrollada también expresamente para este proyecto y que describiremos en el siguiente punto).

min_intensity: Energía mínima (a nivel de fonema, dentro de una palabra):

- Para cada fonema, utilizamos la información de tiempo final y tiempo inicial que hemos obtenido con nuestro programa de búsqueda en lattices HTK.

  Después contrastamos esta información con los valores de energía extraídos a través de Praat [7]. Estos valores son a nivel de frame, no a nivel de fonema, por lo que para calcular el valor de energía de cada fonema hacemos la media de los valores de energía de todos los frames contenidos en el tiempo que dura el fonema. Esa media será nuestro valor de energía para ese fonema.

Finalmente identificamos el valor mínimo de esos valores de energía de fonema con respecto a todos los fonemas que forman la palabra y obtenemos nuestro valor "min_intensity".

mean_intensity: Energía media (a nivel de fonema, dentro de una palabra):

- Para cada fonema, utilizamos la información de tiempo final y tiempo inicial que hemos obtenido con nuestro programa de búsqueda en lattices HTK.

  Después contrastamos esta información con los valores de energía extraídos a través de Praat [7]. Estos valores son a nivel de frame, no a nivel de fonema, por lo que para calcular el valor de energía de cada fonema hacemos la media de los valores de energía de todos los frames contenidos en el tiempo que dura el fonema. Esa media será nuestro valor de energía para ese fonema.

  Finalmente hacemos la media de esos valores por fonema con respecto al número de fonemas que forman la palabra y obtenemos nuestro valor "mean_intensity".

min_pitch_nonzero: Pitch mínimo (a nivel de fonema y distinto de cero)

- Para cada fonema, utilizamos la información de tiempo final y tiempo inicial que hemos obtenido con nuestro programa de búsqueda en lattices HTK.

  Después contrastamos esta información con los valores de pitch extraídos a través de Praat [7]. Estos valores son a nivel de frame, no a nivel de fonema, por lo que para calcular el valor de pitch de cada fonema hacemos la media de los valores de pitch de todos los frames contenidos en el tiempo que dura el fonema. Esa media será nuestro valor de pitch para ese fonema.

  Finalmente identificamos el valor mínimo de esos valores de pitch de fonema con respecto a todos los fonemas que forman la palabra (sin tener en cuenta los valores que sean cero) y obtenemos nuestro valor "min_pitch_nonzero".

min_intensity_nonzero: Energía mínima (a nivel de fonema y distinto de cero)

- Para cada fonema, utilizamos la información de tiempo final y tiempo inicial que hemos obtenido con nuestro programa de búsqueda en lattices HTK.

  Después contrastamos esta información con los valores de energía extraídos a través de Praat [7]. Estos valores son a nivel de frame, no a nivel de fonema, por lo que para calcular el valor de energía de cada fonema hacemos la media de los valores de energía de todos los frames contenidos en el tiempo que dura el fonema. Esa media será nuestro valor de energía para ese fonema.

Finalmente identificamos el valor mínimo de esos valores de energía de fonema con respecto a todos los fonemas que forman la palabra (sin tener en cuenta los valores que sean cero) y obtenemos nuestro valor “min_intensity_nonzero”.

Como resultado obtenemos un fichero donde cada registro nos muestra el nombre de la palabra identificada, el lattice desde donde se extrajo, y todas las características anteriormente descritas.

```
#Extracted     features        based   on      phonemes        information     of      each    word
#word   file    max_ascore_normalizedbyphoneduration    min_ascore_normalizedbyphoneduration    ascore_normalizedbyphoneduration_ave
cual    /home/EAGE0301. 0.0000  -98.3625        -64.7223        0.0000  -3.3100 -1.9350 600000  300000  475000  227.56  190.76  213.
cien    /home/EAGE0301. 0.0000  -95.5000        -65.4708        0.0000  -1.7900 -0.9950 1200000 400000  775000  254.83  188.09  228.
alto    /home/EAGE0301. 0.0000  -87.5400        -46.7280        0.0000  -3.6200 -2.1875 2100000 300000  1000000 256.21  205.99  227.
largo   /home/EAGE0301. 0.0000  -103.7167       -77.2903        0.0000  -3.7000 -2.1680 500000  300000  380000  215.19  0.00    209.
mares   /home/EAGE0301. 0.0000  -97.4586        -67.3326        0.0000  -2.6300 -1.7580 700000  500000  620000  249.82  210.10  231.
mares   /home/EAGE0301. 0.0000  -89.9820        -68.3584        0.0000  -2.6300 -1.7580 1000000 500000  620000  188.09  174.11  180.
cuanto  /home/EAGE0301. 0.0000  -90.2875        -68.2162        0.0000  -2.8200 -1.8383 600000  300000  466666  256.21  190.76  223.
cuenca  /home/EAGE0301. 0.0000  -96.2167        -74.2669        0.0000  -3.6700 -1.7850 600000  300000  466666  256.21  190.76  223.
```

Figura 27. Ejemplo de fichero de características fonéticas extraídas a partir de lattice

4.2.2 Extracción del pitch y energía (a nivel de fonema)

Desarrollamos otro programa (en C) que recibe como argumentos el archivo .mlf a nivel de fonema (extraído a través del programa que desarrollamos para la búsqueda en lattices) y el archivo de salida donde se mostrarán las características prosódicas del conjunto que mencionamos anteriormente (las relacionadas con el pitch y la energía). Para conseguir ese archivo de salida, el programa va comparando cada lattice dentro del fichero .mlf con su archivo de pitch y energía equivalente obtenido a través de PRAAT [7] (fichero de pitch y energía por frame, de un fichero de audio), va calculando las diferentes características de cada fonema y finalmente las agrupa (en máximo valor, mínimo y medio) dentro de una misma palabra y presenta los resultados.

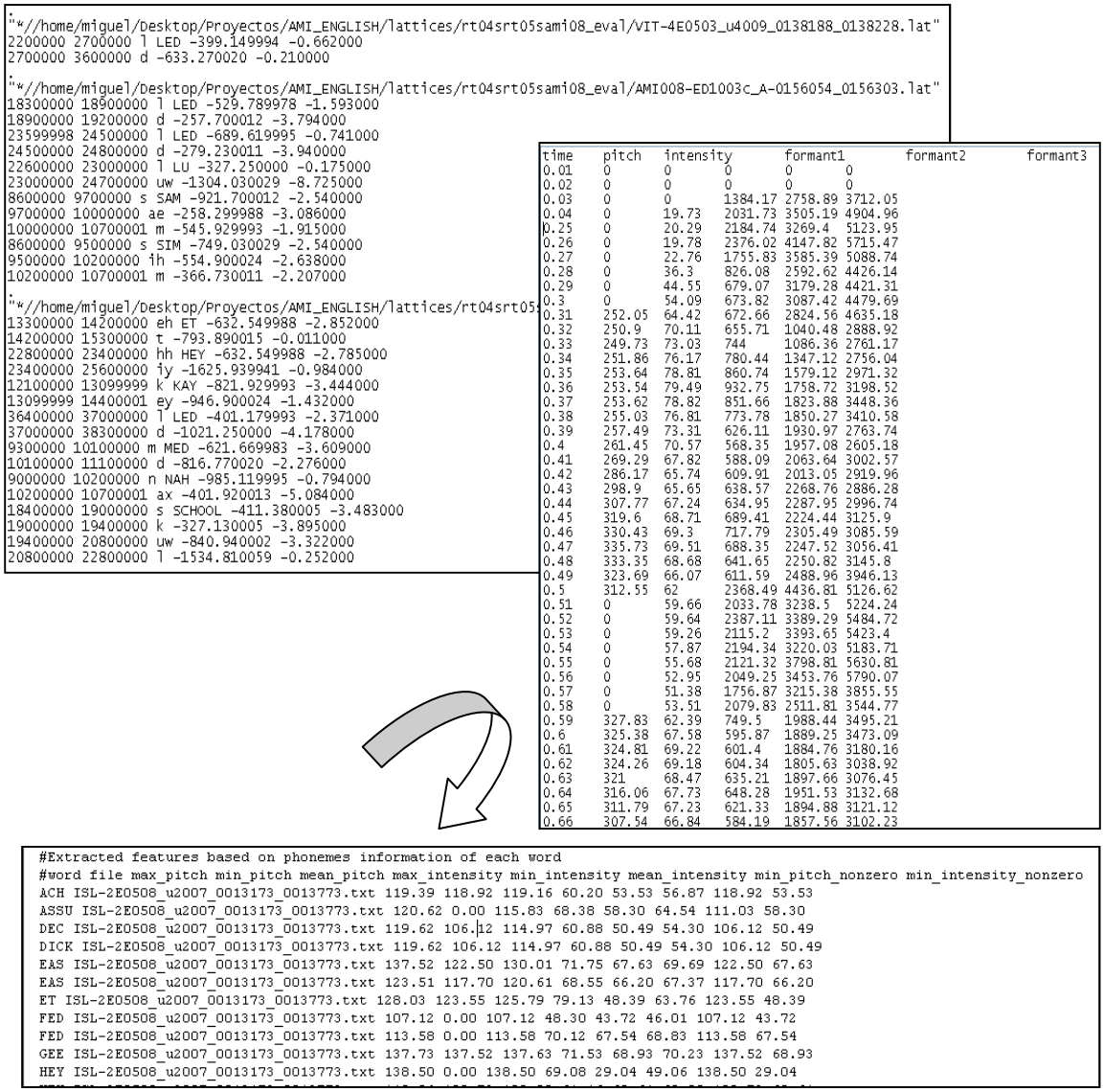


Figura 28. Archivo de características prosódicas obtenido utilizando el archivo .mlf a nivel de fonema y el archivo de pitch y energía extraído con Praat

4.2.3 Pseudocódigo

Main.c

{//inicio de main.c

While (archivo_fonemas) *//Va leyendo línea por línea el archivo.mlf de fonemas*

{//inicio de while (archivo_fonemas)

//El 'punto' indica que ha terminado de leer los datos de un lattice

Nuevo_archivo: if (lectura_fonemas[0] == '.')

{*inicio de if (lectura_fonemas[0]=='.')*

if (num_fonemas!=0) *//Así no entrará en la primera iteración*

{

for (num_fonemas)

{*//Ordena los Pitch y Energías de los fonemas que forman una palabra, de menor a mayor*

mean_phoneme_pitch_vector

mean_phoneme_intensity_vector

}

}

//Guardamos el archivo de pitch y energía en una estructura llamada pitch_structure

While (pitch_file)

{

pitch_structure.pitch[contador] = pitch_aux;

pitch_structure.intensity[contador] = intensity_aux;

contador++;

}

//Reiniciamos el contador de frames del archive de pitch, ya hemos terminado de leer ese archivo

contador = 0;

//Aquí iríamos al inicio del bucle principal (esto sucederá en casos especiales en los que en un archivo no se ha reconocido ninguna palabra)

if (lectura_fonemas[0]=='.')

goto **Nuevo_archivo;**

//Leemos los datos del .mlf

fscanf(archivo_fonemas, "%s", lectura_fonemas); *//tiempo inicial*

tiempoinicial_aux = atoi(lectura_fonemas);*//Guarda el tiempo inicial*

fscanf(archivo_fonemas, "%s", lectura_fonemas); *//tiempo final*

tiempofinal_aux = atoi(lectura_fonemas*);//Guarda el tiempo final*

fscanf(archivo_fonemas, "%s", lectura_fonemas*);//fonema*

fscanf(archivo_fonemas, "%s", lectura_fonemas*); //WORD*

```
//Escribimos la palabra en el archivo de salida
fprintf(data_file,"\n%s",lectura_fonemas);

//Escribimos el nombre de archivo en el fichero de salida
fprintf(data_file," %s",nombre_archivo);

//Dividimos entre 100000 para trabajar con una posición entera, dentro de
los vectores de pitch y de intensity de la "pitch_structure"
initial_frame_position = tiempoinicial_aux/100000;
final_frame_position = tiempofinal_aux/100000;

for (i=initial_frame_position;i<final_frame_position;i++)
{
//Guardamos los valores de pitch y de energía de todos los frames distintos
de '0' que estén contenidos en el tiempo que dura el fonema …
  if(pitch_structure.pitch[i]!=0)
 {
    pitch[pitch_frames_number]=pitch_structure.pitch[i];
    mean_pitch = mean_pitch + pitch[pitch_frames_number];
 }
  if(pitch_structure.intensity[i]!=0)
  {
  intensity[intensity_frames_number]=pitch_structure.intensity[i];
  mean_intensity=mean_intensity+
  intensity[intensity_frames_number];
  }
 pitch_frames_number++ //aumenta el número de frames, si no es 0
 intensity_frames_number++//aumenta el número de frames, si no es 0
}

//… y calculamos la media
mean_pitch=mean_pitch/pitch_frames_number;
mean_intensity=mean_intensity/intensity_frames_number;

//Actualizamos el valor medio de pitch y de energía para la Palabra
(finalmente lo dividiremos entre el número de fonemas)
mean_phoneme_pitch_vector[num_fonemas] = mean_pitch;
mean_phoneme_intensity_vector[num_fonemas] = mean_intensity;

word_mean_pitch =
word_mean_pitch+mean_phoneme_pitch[num_fonemas];

word_mean_intensity =
word_mean_intensity+mean_phoneme_intensity [num_fonemas];


//Actualizamos el número de fonemas sin pitch cero
if (mean_phoneme_pitch_vector[num_fonemas]!=0)
        num_fonemas_sin_pitch_cero++;
```

```
        //Actualizamos el número de fonemas sin energía cero
        if (mean_phoneme_intensity_vector[num_fonemas]!=0)
                num_fonemas_sin_intensity_cero++;


        //Cada línea que vamos leyendo del archivo_fonemas es un fonema
        num_fonemas++;

} fin de if (lectura_fonemas[0]=='.')

else //..if (lectura_fonemas[0] != '.')
{
        //Cuando encontramos una nueva palabra calculamos los valores de la
        anterior
        If ("nueva_palabra")
        {
                for (num_fonemas)
                {//Ordena los Pitch y Energías de los fonemas que forman una
                palabra, de menor a mayor
                        mean_phoneme_pitch_vector
                        mean_phoneme_intensity_vector
                }



                //Pitch máximo y mínimo de la palabra
                word_max_pitch=
                mean_phoneme_pitch_vector[num_fonemas-1];
                word_min_pitch = mean_phoneme_pitch_vector[0];

                //Pitch mínimo distinto de cero
                for(i=0;i<=num_fonemas;i++)
                {
                  word_min_pitch_nonzero =
                  mean_phoneme_pitch_vector[i];
                  if(word_min_pitch_nonzero!=0)
                  break;
                }

                //Energía máxima y mínima de la palabra
                word_max_intensity=
                mean_phoneme_intensity_vector[num_fonemas-1];
                word_min_intensity = mean_phoneme_intensity_vector[0];

                //Energía mínima distinta de cero
                for(i=0;i<=num_fonemas;i++))
                {
                  word_min_intensity_nonzero =
                  mean_phoneme_pitch_vector[i];
                  if(word_min_pitch_nonzero!=0)
                  break;
                }
```

```
        //Pitch medio de la palabra
        word_mean_pitch =
        word_mean_pitch/num_fonemas_sin_pitch_cero;

        //Energía Media de la palabra
        word_mean_intensity =
        word_mean_intensity/num_fonemas_sin_intensity_cero;



        //Escribimos las características de la palabra anterior
        fprintf(data_file," %.2lf %.2lf %.2lf %.2lf %.2lf %.2lf %.2lf
        %.2lf",word_max_pitch,word_min_pitch,word_mean_pitch,
        word_max_intensity,word_min_intensity,word_mean_intensi
        ty,word_min_pitch_nonzero,word_min_intensity_nonzero);

        //Reiniciamos el num_fonemas
        num_fonemas = 0;
        num_fonemas_sin_pitch_cero = 0;
        num_fonemas_sin_intensity_cero = 0;

}//fin del if (nueva palabra)


//Dividimos entre 100000 para trabajar con una posición entera, dentro de
los vectores de pitch y de intensity de la "pitch_structure"
initial_frame_position = tiempoinicial_aux/100000;
final_frame_position = tiempofinal_aux/100000;

for (i=initial_frame_position;i<final_frame_position;i++)
{
//Guardamos los valores de pitch y de energía de todos los frames distintos
de '0' que estén contenidos en el tiempo que dura el fonema…
  if(pitch_structure.pitch[i]!=0)
 {
    pitch[pitch_frames_number]=pitch_structure.pitch[i];
    mean_pitch = mean_pitch + pitch[pitch_frames_number];
 }
  if(pitch_structure.intensity[i]!=0)
  {
  intensity[intensity_frames_number]=pitch_structure.intensity[i];
  mean_intensity=mean_intensity+
  intensity[intensity_frames_number];
   }

 pitch_frames_number++ //aumenta el número de frames, si no es 0
 intensity_frames_number++//aumenta el número de frames, si no es 0

}
//… y calculamos la media
```

```
                    mean_pitch=mean_pitch/pitch_frames_number;
                    mean_intensity=mean_intensity/intensity_frames_number;

                    //Vamos actualizando el valor medio de pitch y de energía para la Palabra
                    (finalmente lo dividiremos entre el número de fonemas)
                    mean_phoneme_pitch_vector[num_fonemas] = mean_pitch;
                    mean_phoneme_intensity_vector[num_fonemas] = mean_intensity;

                    word_mean_pitch = word_mean_pitch +
                    mean_phoneme_pitch_vector[num_fonemas];


                    word_mean_intensity = word_mean_intensity +
                    mean_phoneme_intensity_vector[num_fonemas];

                    //Actualizamos el número de fonemas sin pitch cero
                    if (mean_phoneme_pitch_vector[num_fonemas]!=0)
                            num_fonemas_sin_pitch_cero++;

                    //Actualizamos el número de fonemas sin pitch cero
                    if (mean_phoneme_intensity_vector[num_fonemas]!=0)
                            num_fonemas_sin_intensity_cero++;



                    //Siguiente fonema
                    num_fonemas++;

            {//fin de else //..if (lectura_fonemas[0] != '.')
        }//fin de while (archivo_fonemas)

}//fin de main.c
```

5

PRUEBAS Y RESULTADOS

5.1 Búsqueda en Lattices

5.1.1 ALBAYZIN

```
#!MLF!#
"*//EAGE0301.rec"
3600000 5500000 cual
18500000 21600000 cien
2400000 6400000 alto
10500000 12400000 largo
6800000 9900000 mares
16000000 19100000 mares
3600000 6400000 cuanto
3600000 6400000 cuenca
.
"*//EAGE0349.rec"
12600000 15100000 pasa
12500000 13900000 cual
15000000 16300000 cual
4200000 5800000 esta
13300000 15000000 esta
18800000 22100000 cien
12600000 15100000 pase
6000000 8800000 llama
6000000 9200000 llaman
```

Ejecutamos nuestra herramienta de búsqueda en lattices customizada para el formato de HTK, contra varios conjuntos de datos de ALBAYZIN: Una ejecución sobre 4400 lattices de TRAIN utilizando un diccionario de 105 palabras (TRAIN); otra ejecución contra los mismos lattices de TRAIN pero esta vez utilizando un diccionario de 500 palabras (DEV) y una última ejecución contra un conjunto de 2400 lattices de TEST utilizando un diccionario de 400 palabras (TEST).

Como resultado obtenemos un archivo .mlf ("master label file") a nivel de palabra para cada ejecución.

Figura 29. Archivo .mlf ("master label file") a nivel de palabra

También obtenemos un archivo .mlf a nivel de fonema, esto es, un archivo con el mismo formato que el anterior con el añadido de que tras mostrar el tiempo inicial y final de la palabra va presentando la información para cada fonema: el tiempo inicial, tiempo final, score acústico y de modelo de lenguaje de cada fonema incluido dentro de la palabra. Cabe destacar que este tipo de fichero no había sido generado en el pasado, lo que lo hace interesante a la hora de tener ya aislados los fonemas incluidos dentro de una palabra que ha sido detectada en el lattice y poder trabajar con ellos para extraer más características a ese nivel (de fonema).

```
#!MLF!#
"*//home/miguel/Desktop/lattices_material/latticesTRAIN/EAGE0301.lat"
3600000 4200000 k cual 0.000000 0.000000
4200000 4500000 w -241.919998 -2.820000
4500000 5100000 a -479.320007 -1.610000
5100000 5500000 l -393.450012 -3.310000
18500000 19400000 T cien 0.000000 0.000000
19400000 19800000 j -308.220001 -1.400000
19800000 20400000 E -535.969971 -0.790000
20400000 21600000 n -1146.000000 -1.790000
2400000 3400000 A alto 0.000000 0.000000
3400000 5500000 l -391.549988 -3.070000
5500000 5800000 t -262.619995 -3.620000
5800000 6400000 o -484.359985 -2.060000
10500000 11000000 l largo 0.000000 0.000000
11000000 11300000 A -262.500000 -3.040000
11300000 11700000 r -383.809998 -1.660000
11700000 12000000 G -311.149994 -3.700000
12000000 12400000 o -397.130005 -2.440000
6800000 7300000 m mares 0.000000 0.000000
7300000 8000000 A -564.510010 -2.480000
8000000 8600000 r -503.429993 -1.660000
8600000 9200000 e -447.929993 -2.630000
9200000 9900000 s -682.210022 -2.020000
16000000 16500000 m mares 0.000000 0.000000
16500000 17000000 A -425.179993 -2.480000
17000000 17500000 r -449.910004 -1.660000
17500000 18500000 e -818.489990 -2.630000
18500000 19100000 s -509.549988 -2.020000
3600000 4200000 k cuanto 0.000000 0.000000
4200000 4500000 w -241.919998 -2.820000
4500000 5100000 a -440.880005 -1.610000
5100000 5500000 n -361.149994 -2.670000
5500000 5800000 t -252.490005 -1.870000
5800000 6400000 o -484.359985 -2.060000
3600000 4200000 k cuenca 0.000000 0.000000
4200000 4600000 w -325.709991 -2.820000
4600000 5100000 E -444.940002 -0.560000
5100000 5400000 n -288.649994 -1.790000
5400000 5800000 k -363.029999 -3.670000
5800000 6400000 a -529.270020 -1.870000
.
"*//home/miguel/Desktop/lattices_material/latticesTRAIN/EAGE0349.lat"
12600000 13099999 p pasa 0.000000 0.000000
13099999 13700000 A -479.470001 -2.830000
```

Figura 30. Archivo .mlf a nivel de fonema

5.1.2 AMI

Ejecutamos el programa de búsqueda en lattices dos veces contra un conjunto de 1758 lattices, utilizando un diccionario de 490 palabras y otro diccionario de 67 palabras (Conjunto "TRAIN" – formado por palabras *inv** y *oov* ** – y Conjunto "DEV" – formado por palabras *oov*** – respectivamente).

También ejecutamos el programa otras dos veces, contra un conjunto de 11092 lattices de TEST, utilizando un diccionario de 268 palabras y otro diccionario de 484 palabras (Conjunto "INV" – formado por palabras *inv** y conjunto OOV – formado por palabras *oov* ** – respectivamente).

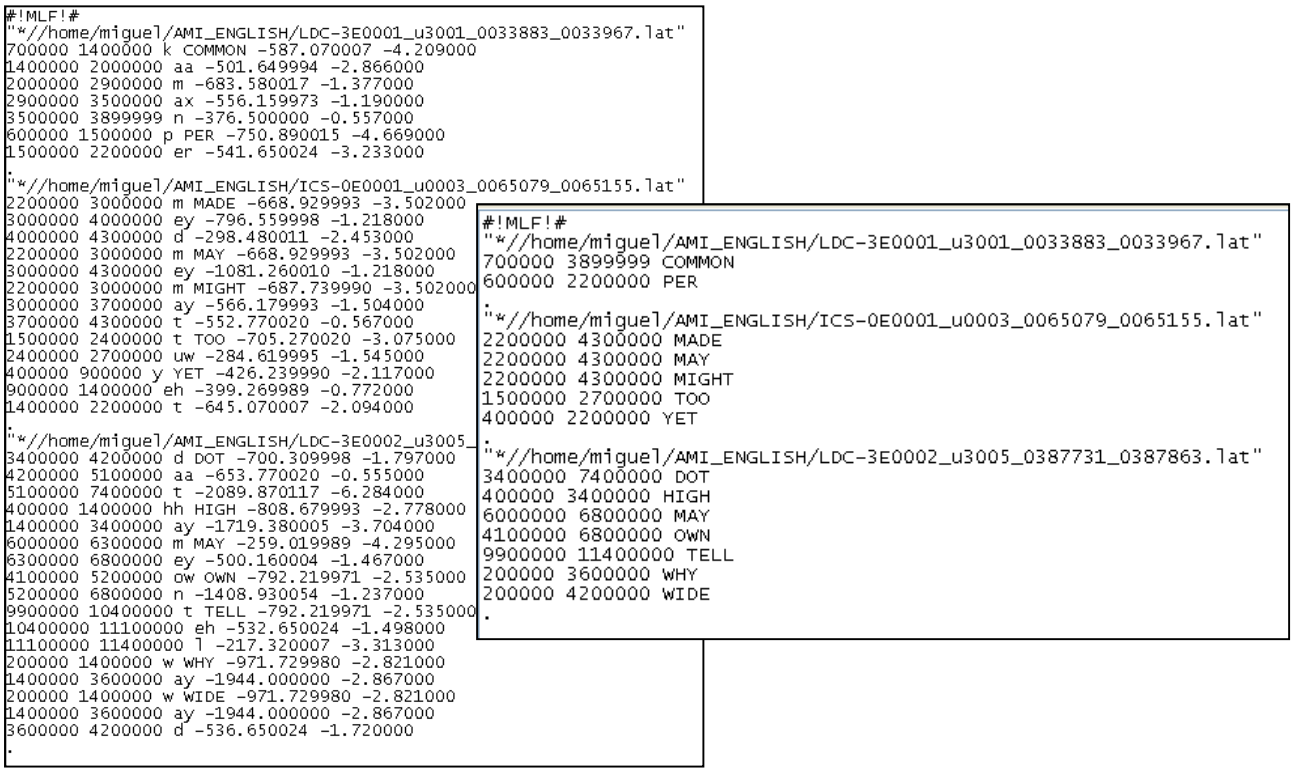


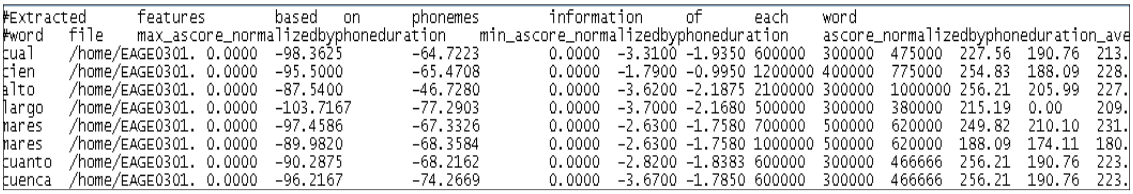
Figura 31. Archivos .mlf a nivel de fonema y a nivel de palabra (AMI)

**inv:* palabras cuya transcripción fonética no tiene errores.
***oov:* palabras cuya transcripción fonética tiene errores.

5.2 Extracción de Características y Cálculo del Pitch y Energía

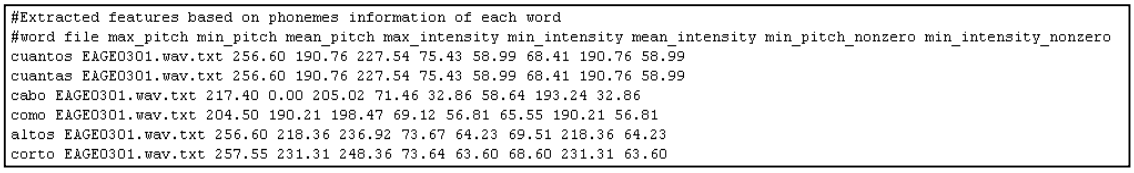
Ejecutamos el programa de cálculo de características prosódicas y fonéticas que hemos desarrollado contra las bases de datos de ALBAYZIN y AMI.

5.2.1 ALBAYZIN



#Extracted #word	features file	based on max_ascore_normalizedbyphoneduration		phonemes min_ascore_normalizedbyphoneduration	information	of	each	word ascore_normalizedbyphoneduration_ave				
cual	/home/EAGE0301.	0.0000	-98.3625	-64.7223	0.0000	-3.3100	-1.9350	600000	300000	475000	227.56	190.76	213.
cien	/home/EAGE0301.	0.0000	-95.5000	-65.4708	0.0000	-1.7900	-0.9950	1200000	400000	775000	254.83	188.09	228.
alto	/home/EAGE0301.	0.0000	-87.5400	-46.7280	0.0000	-3.6200	-2.1875	2100000	300000	1000000	256.21	205.99	227.
largo	/home/EAGE0301.	0.0000	-103.7167	-77.2903	0.0000	-3.7000	-2.1680	500000	300000	380000	215.19	0.00	209.
mares	/home/EAGE0301.	0.0000	-97.4586	-67.3326	0.0000	-2.6300	-1.7580	700000	500000	620000	249.82	210.10	231.
mares	/home/EAGE0301.	0.0000	-89.9820	-68.3584	0.0000	-2.6300	-1.7580	1000000	500000	620000	188.09	174.11	180.
cuanto	/home/EAGE0301.	0.0000	-90.2875	-68.2162	0.0000	-2.8200	-1.8383	600000	300000	466666	256.21	190.76	223.
cuenca	/home/EAGE0301.	0.0000	-96.2167	-74.2669	0.0000	-3.6700	-1.7850	600000	300000	466666	256.21	190.76	223.

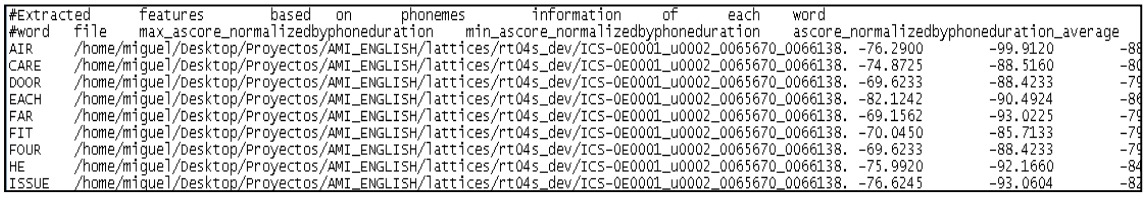
Figura 32. Ejemplo de fichero de características fonéticas extraídas a partir de lattice (ALBAYZIN)



```
#Extracted features based on phonemes information of each word
#word file max_pitch min_pitch mean_pitch max_intensity min_intensity mean_intensity min_pitch_nonzero min_intensity_nonzero
cuantos EAGE0301.wav.txt 256.60 190.76 227.54 75.43 58.99 68.41 190.76 58.99
cuantas EAGE0301.wav.txt 256.60 190.76 227.54 75.43 58.99 68.41 190.76 58.99
cabo EAGE0301.wav.txt 217.40 0.00 205.02 71.46 32.86 58.64 193.24 32.86
como EAGE0301.wav.txt 204.50 190.21 198.47 69.12 56.81 65.55 190.21 56.81
altos EAGE0301.wav.txt 256.60 218.36 236.92 73.67 64.23 69.51 218.36 64.23
corto EAGE0301.wav.txt 257.55 231.31 248.36 73.64 63.60 68.60 231.31 63.60
```

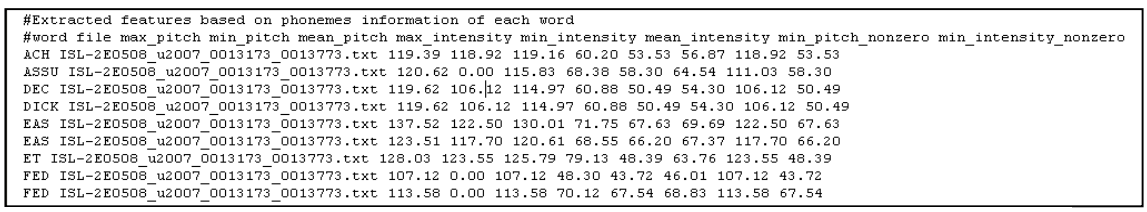
Figura 33. Ejemplo de fichero de salida, tras el cálculo de las nuevas características prosódicas a nivel de fonema (ALBAYZIN)

5.2.2 AMI



#Extracted #word	features file	based on max_ascore_normalizedbyphoneduration	phonemes min_ascore_normalizedbyphoneduration	information	of	each	word ascore_normalizedbyphoneduration_average	
AIR	/home/miguel/Desktop/Proyectos/AMI_ENGLISH/lattices/rt04s_dev/ICS-0E0001_u0002_0065670_0066138.	-76.2900	-99.9120					-88
CARE	/home/miguel/Desktop/Proyectos/AMI_ENGLISH/lattices/rt04s_dev/ICS-0E0001_u0002_0065670_0066138.	-74.8725	-88.5160					-80
DOOR	/home/miguel/Desktop/Proyectos/AMI_ENGLISH/lattices/rt04s_dev/ICS-0E0001_u0002_0065670_0066138.	-69.6233	-88.4233					-79
EACH	/home/miguel/Desktop/Proyectos/AMI_ENGLISH/lattices/rt04s_dev/ICS-0E0001_u0002_0065670_0066138.	-82.1242	-90.4924					-86
FAR	/home/miguel/Desktop/Proyectos/AMI_ENGLISH/lattices/rt04s_dev/ICS-0E0001_u0002_0065670_0066138.	-69.1562	-93.0225					-79
FIT	/home/miguel/Desktop/Proyectos/AMI_ENGLISH/lattices/rt04s_dev/ICS-0E0001_u0002_0065670_0066138.	-70.0450	-85.7133					-79
FOUR	/home/miguel/Desktop/Proyectos/AMI_ENGLISH/lattices/rt04s_dev/ICS-0E0001_u0002_0065670_0066138.	-69.6233	-88.4233					-79
HE	/home/miguel/Desktop/Proyectos/AMI_ENGLISH/lattices/rt04s_dev/ICS-0E0001_u0002_0065670_0066138.	-75.9920	-92.1660					-84
ISSUE	/home/miguel/Desktop/Proyectos/AMI_ENGLISH/lattices/rt04s_dev/ICS-0E0001_u0002_0065670_0066138.	-76.6245	-93.0604					-82

Figura 34. Ejemplo de fichero de características fonéticas extraídas a partir de lattice (AMI)



```
#Extracted features based on phonemes information of each word
#word file max_pitch min_pitch mean_pitch max_intensity min_intensity mean_intensity min_pitch_nonzero min_intensity_nonzero
ACH ISL-2E0508_u2007_0013173_0013773.txt 119.39 118.92 119.16 60.20 53.53 56.87 118.92 53.53
ASSU ISL-2E0508_u2007_0013173_0013773.txt 120.62 0.00 115.83 68.38 58.30 64.54 111.03 58.30
DEC ISL-2E0508_u2007_0013173_0013773.txt 119.62 106.12 114.97 60.88 50.49 54.30 106.12 50.49
DICK ISL-2E0508_u2007_0013173_0013773.txt 119.62 106.12 114.97 60.88 50.49 54.30 106.12 50.49
EAS ISL-2E0508_u2007_0013173_0013773.txt 137.52 122.50 130.01 71.75 67.63 69.69 122.50 67.63
EAS ISL-2E0508_u2007_0013173_0013773.txt 123.51 117.70 120.61 68.55 66.20 67.37 117.70 66.20
ET ISL-2E0508_u2007_0013173_0013773.txt 128.03 123.55 125.79 79.13 48.39 63.76 123.55 48.39
FED ISL-2E0508_u2007_0013173_0013773.txt 107.12 0.00 107.12 48.30 43.72 46.01 107.12 43.72
FED ISL-2E0508_u2007_0013173_0013773.txt 113.58 0.00 113.58 70.12 67.54 68.83 113.58 67.54
```

Figura 35. Ejemplo de fichero de salida, tras el cálculo de las nuevas características prosódicas a nivel de fonema (AMI)

5.3 Análisis de Resultados

Para llevar a cabo el análisis de resultados hemos utilizado los métodos de análisis incremental de varianza, evaluación individual de características (a partir del evaluador GainRatio) y clasificación del conjunto de características más relevantes a través de un Árbol de decisión.

5.3.1 ALBAYZIN

5.3.1.1 Análisis Incremental de la Varianza

En la Tabla 2 podemos ver el resultado del Análisis Incremental añadiendo paso a paso los diferentes conjuntos de características evaluados en pasados estudios (que ya presentamos en la sección de "Medios Disponibles y Diseño"), sobre la base de datos de ALBAYZIN [4]. Se puede ver que se llega a explicar un 60.5649% de Varianza.

Feature Sets	R^2(%)	R^2 Increment (%)
LAT	52,3077	52,3077
+DUR	59,3618	7,0541
+PROS	60,0592	0,6974
+LEX	60,4241	0,3649
+LEV	60,5647	0,1406
+POS	60,5649	0,0002

Tabla 2. Análisis de Regresión Lineal. Los conjuntos de características se van añadiendo en el orden que maximiza la varianza explicada. Se muestra la varianza explicada en porcentaje y el incremento absoluto con respecto al último conjunto de características añadido.

Como algunas de las características mostradas en la Tabla 2 dentro de un mismo grupo pueden ser redundantes, es necesario hacer un análisis añadiendo cada característica de forma individual, y no por grupos. De este modo, a través de un Análisis Incremental de la Varianza (incluyendo ya las características a nivel de fonema que hemos extraído en este proyecto), vemos que algunas de las nuevas características se han colocado en las primeras posiciones. En concreto se ha colocado en 4ª posición la característica 31 (mínima duración por fonema), en 5ª la 26 (score acústico medio normalizado por duración del fonema) y en 7ª la 30 (máxima duración por fonema), destacamos también la 25 (mínimo score acústico normalizado por duración del fonema) y la 36 (máxima energía de un fonema). (datos de TRAIN, ver Tabla 4).

También hay que destacar que ahora, con el modelo completo, explicamos el 61.7% de la varianza, como podemos ver en la Tabla 4 (en datos de TRAIN). En las pruebas anteriores, sin las nuevas características, explicábamos sólo el 60.56%. Puede parecer poco un 1.14% más, pero es muy significativo porque cuando tienes muchas características subir un poco más el porcentaje de varianza explicada es muy importante.

1	score
2	effective_hit_rate
3	effective_FA_rate
4	duration
5	number_graphemes
6	number_phones
7	number_vowels
8	number_consonants
9	number_phone_vowels
10	number_phone_consonants
11	speaker_phone_rate
12	speaker_vowel_rate
13	position
14	max_Levenshtein_distance
15	min_Levenshtein_distance
16	mean_Levenshtein_distance
17	max_pitch
18	min_pitch
19	mean_pitch
20	max_energy
21	min_energy
22	mean_energy
23	voicing_percentage
24	max_acoustic_score_normalizedbyphoneduration
25	min_acoustic_score_normalizedbyphoneduration
26	mean_acoustic_score_normalized_byphoneduration
27	max_languagemodelscore
28	min_languagemodelscore
29	mean_languagemodelscore
30	max_duration
31	min_duration
32	mean_duration
33	max_pitch_phoneme
34	min_pitch_phoneme
35	mean_pith_phoneme
36	max_energy_phoneme
37	min_energy_phoneme
38	mean_energy_phoneme
39	min_pitch_phoneme_nonzero
40	min_energy_phoneme_nonzero

Tabla 3. Lista completa de características

En las páginas siguientes podemos ver el resumen de este Análisis Incremental de la Varianza, generando un modelo completo con todas las variables añadiendo una a una la que más varianza residual explica:

Analizing TRAIN balanced data adding the feature that maximices R2		
FEATURE	NAME	R2
1	score	0.412282
4	duration	0.548585
3	effective_FA_rate	0.572024
31	**min_duration**	0.585926
26	**mean_acoustic_score_normalized_byphoneduration**	0.594215
2	effective_hit_rate	0.597929
30	**max_duration**	0.601413
23	voicing_percentage	0.604494
25	**min_acoustic_score_normalizedbyphoneduration**	0.605902
36	**max_energy_phoneme**	0.607361
12	speaker_vowel_rate	0.609028
9	number_phone_vowels	0.610326
15	min_Levenshtein_distance	0.611244
16	mean_Levenshtein_distance	0.612193
21	min_energy	0.612718
40	min_energy_phoneme_nonzero	0.614032
37	min_energy_phoneme	0.614485
38	mean_energy_phoneme	0.614909
7	number_vowels	0.615226
29	mean_languagemodelscore	0.615449
14	max_Levenshtein_distance	0.615633
28	min_languagemodelscore	0.615832
24	max_acoustic_score_normalizedbyphoneduration	0.616052
20	max_energy	0.616163
17	max_pitch	0.616283
13	position	0.616391
6	number_phones	0.616475
5	number_graphemes	0.616684
34	min_pitch_phoneme	0.616724
22	mean_energy	0.616757
18	min_pitch	0.616795
39	min_pitch_phoneme_nonzero	0.616828
35	mean_pitch_phoneme	0.616837
19	mean_pitch	0.616898
33	max_pitch_phoneme	0.616909
11	speaker_phone_rate	0.616916
32	mean_duration	0.616989
8	number_consonants	0.616995
10	number_phone_consonants	0.616995
27	max_languagemodelscore	**0.616995**

Tabla 4. Análisis Incremental de la Varianza para el conjunto de datos TRAIN de ALBAYZIN

Analizing DEV prosodic balanced data adding the feature that maximices R2		
FEATURE	NAME	R2
1	score	0.359390
4	duration	0.486582
29	mean_languagemodelscore	0.504200
14	max_Levenshtein_distance	0.518361
28	min_languagemodelscore	0.526206
13	position	0.532795
31	min_duration	0.536812
10	number_phone_consonants	0.539893
21	min_energy	0.541727
40	min_energy_phoneme_nonzero	0.542680
2	effective_hit_rate	0.543580
3	effective_FA_rate	0.545062
38	mean_energy_phoneme	0.546021
22	mean_energy	0.546587
12	speaker_vowel_rate	0.546909
7	number_vowels	0.548221
30	max_duration	0.549081
35	mean_pitch_phoneme	0.549398
34	min_pitch_phoneme	0.550074
39	min_pitch_phoneme_nonzero	0.550326
33	max_pitch_phoneme	0.550648
15	min_Levenshtein_distance	0.550849
26	mean_acoustic_score_normalized_byphoneduration	0.550956
25	min_acoustic_score_normalizedbyphoneduration	0.551222
27	max_languagemodelscore	0.551474
16	mean_Levenshtein_distance	0.551582
36	max_energy_phoneme	0.551670
20	max_energy	0.552127
37	min_energy_phoneme	0.552202
23	voicing_percentage	0.552231
18	min_pitch	0.552256
32	mean_duration	0.552278
11	speaker_phone_rate	0.552683
6	number_phones	0.552708
17	max_pitch	0.552723
19	mean_pitch	0.552732
24	max_acoustic_score_normalizedbyphoneduration	0.552738
5	number_graphemes	0.552741
8	number_consonants	0.552741
9	number_phone_vowels	0.552741

Tabla 5. Análisis Incremental de la Varianza para el conjunto de datos DEV de ALBAYZIN

Analizing TEST prosodic balanced data adding the feature that maximices R2		
FEATURE	NAME	R2
1	score	0.396083
4	duration	0.526788
3	effective_FA_rate	0.539162
31	min_duration	0.548836
13	position	0.555775
26	mean_acoustic_score_normalized_byphoneduration	0.561859
29	mean_languagemodelscore	0.564673
2	effective_hit_rate	0.566646
10	number_phone_consonants	0.568560
25	min_acoustic_score_normalizedbyphoneduration	0.569880
28	min_languagemodelscore	0.570643
21	min_energy	0.571331
32	mean_duration	0.571940
30	max_duration	0.572938
40	min_energy_phoneme_nonzero	0.573349
6	number_phones	0.573711
12	speaker_vowel_rate	0.574209
23	voicing_percentage	0.574485
16	mean_Levenshtein_distance	0.574739
36	max_energy_phoneme	0.574872
38	mean_energy_phoneme	0.576049
20	max_energy	0.576566
27	max_languagemodelscore	0.576650
37	min_energy_phoneme	0.576705
11	speaker_phone_rate	0.576775
17	max_pitch	0.576817
18	min_pitch	0.577068
19	mean_pitch	0.577116
34	min_pitch_phoneme	0.577153
7	number_vowels	0.577190
14	max_Levenshtein_distance	0.577220
39	min_pitch_phoneme_nonzero	0.577241
22	mean_energy	0.577260
33	max_pitch_phoneme	0.577280
15	min_Levenshtein_distance	0.577297
5	number_graphemes	0.577298
8	number_consonants	0.577299
9	number_phone_vowels	0.577299
24	max_acoustic_score_normalizedbyphoneduration	0.577299
35	mean_pitch_phoneme	0.577299

Tabla 6. Análisis Incremental de la Varianza para el conjunto de datos TEST de ALBAYZIN

Analizando los resultados reflejados en la Tabla 4 debemos destacar que con 9 características ya se consigue explicar el mismo porcentaje de varianza que el mostrado en la Tabla 2, donde hay muchas más características (concretamente 23).

Observamos que las características relacionadas con el lattice y las duraciones son importantes a la hora de discriminar entre acierto y falsa alarma. Éstas ocupan las 7 primeras posiciones en el análisis de varianza incremental (ver Tabla 4. Datos de TRAIN). Las basadas en el lattice, por su intrínseca propiedad de reflejar la confianza del reconocedor en la palabra, así como las duraciones, que tienden a tener valores extremos para falsas alarmas, son las que mejor discriminan entre ambas clases.

Las características prosódicas no aportan apenas información (pitch y energía) para diferenciar entre acierto y falsa alarma, así como algunas léxicas, quizás por su alta correlación con las características relacionadas con la duración (palabras más largas tienden a durar más y las palabras más cortas o con menos fonemas tienden a tener una duración menor). Este comportamiento se observa tanto en los datos de TRAIN, como DEV y TEST.

5.3.1.2 Evaluación de Atributos utilizando WEKA

Utilizamos el Evaluador de Atributos "GainRatioAttributeEval", que evalúa a un atributo (característica) tras medir la tasa de ganancia con respecto a la clase (hit o FA –acierto o falsa alarma–):

GainR(Class, Attribute)=(H(Class) - H(Class | Attribute)) / H(Attribute) *(17)*

ALBAYZIN_Train_Ranker_GainRatio_hitFABalanced		
Ranked attributes		
0.10849	1	score
0.09807	24	max_acoustic_score_normalizedbyphoneduration
0.09234	27	max_languagemodelscore
0.08465	4	duration
0.08089	10	number_phone_consonants
0.08079	3	effective_FA_rate
0.07823	29	mean_languagemodelscore
0.07683	16	mean_Levenshtein_distance
0.07582	2	effective_hit_rate
0.07179	9	number_phone_vowels
0.07052	6	number_phones
0.06235	8	number_consonants
0.06082	28	min_languagemodelscore
0.06062	31	min_duration
0.05929	5	number_graphemes
0.05554	7	number_vowels
0.05419	12	speaker_vowel_rate
0.05221	11	speaker_phone_rate
0.0514	32	mean_duration
0.0426	30	max_duration
0.03952	25	min_acoustic_score_normalizedbyphoneduration
0.03806	26	mean_acoustic_score_normalized_byphoneduration
0.03165	22	mean_energy
0.03098	21	min_energy
0.02964	13	position
0.02524	23	voicing_percentage
0.02391	37	min_energy_phoneme
0.02343	40	min_energy_phoneme_nonzero
0.02322	14	max_Levenshtein_distance
0.01301	19	mean_pitch
0.01238	34	min_pitch_phoneme
0.01096	33	max_pitch_phoneme
0.01025	15	min_Levenshtein_distance
0.01009	38	mean_energy_phoneme
0.00994	17	max_pitch
0.00756	35	mean_pitch_phoneme
0.00746	18	min_pitch
0.00636	39	min_pitch_phoneme_nonzero

ALBAYZIN_Train_Ranker_GainRatio_hitFABalanced		
Ranked attributes		
0.00567	36	max_energy_phoneme
0.00457	20	max_energy

Tabla 7. Evaluación de Características GainRatio en conjunto de datos TRAIN (ALBAYZIN)

En este caso, analizamos cada característica de forma individual, mediante la ganancia que proporciona a la clase. De nuevo podemos ver como las características relacionadas con el lattice y las duraciones son las más importantes, quedando en las primeras posiciones. En este caso, existen también características léxicas y relacionadas con la distancia de Levenshtein que quedan en posiciones superiores. Esto se debe al hecho de que palabras más cortas (con menos fonemas) son más susceptibles de error, mientras que palabras con distancia de Levenshtein similar tienen más probabilidad de ser confundidas a la hora de realizar el reconocimiento de voz. De nuevo, las características relacionadas con el pitch y la energía quedan en las posiciones más bajas, mostrándonos que apenas son informativas para diferenciar acierto y falsa alarma.

5.3.1.3 Clasificación a través de Árbol de Decisión

Utilizamos también la herramienta WEKA (Herramienta de Aprendizaje Automático) para hacer una clasificación de los atributos (basados en la evaluación con GainRatio) a través de un Árbol de Decisión (proporcionándole el conjunto de datos TRAIN como datos de entrenamiento y el conjunto TEST como datos de test).

Podemos ver, bajo estas líneas, el resumen de los resultados obtenidos, tras ejecutar varias veces el Árbol de Decisión. Para cada ejecución tuvimos que preparar un conjunto de datos de entrenamiento y otro de test con los datos de las características que queríamos evaluar. En cada ejecución fuimos añadiendo progresivamente la característica/s con mayor valor en la evaluación anterior:

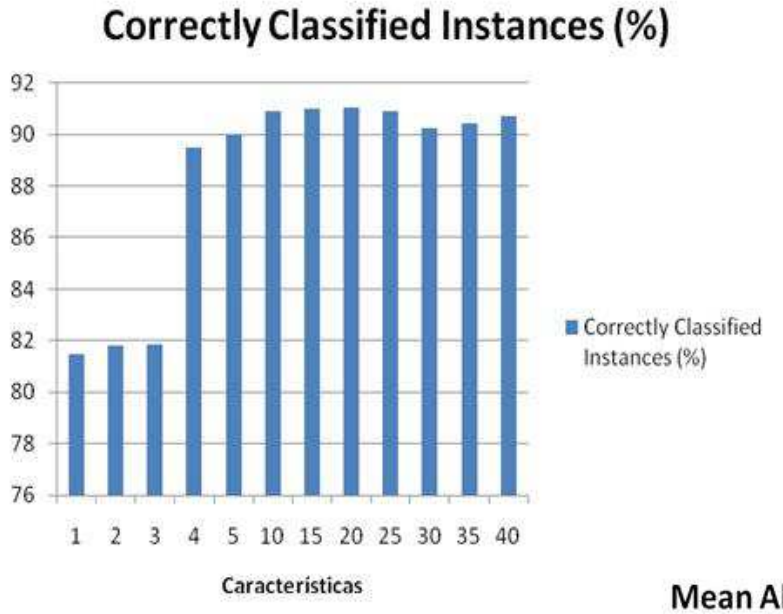


Figura 36. Porcentaje de ocurrencias correctamente clasificadas (ALBAYZIN)

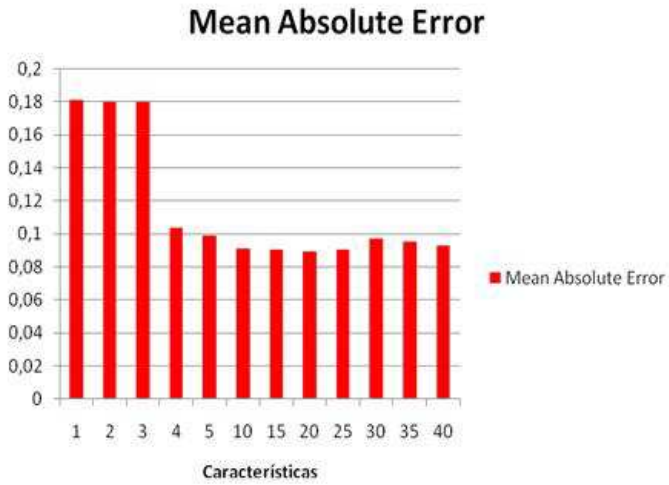


Figura 37. Tasa de Error Absoluto de Clasificación (ALBAYZIN). Nos da la medida de las diferencias en promedio entre los valores pronosticados y los observados

Aquí podemos ver como el hecho de añadir las características más significativas derivadas de los análisis anteriores (en concreto del análisis basado en GainRatio), proporciona una mejor tasa de clasificación que el uso de la mejor característica de forma aislada.

Sin embargo, a partir de cierto punto la tasa de clasificación empeora ligeramente, lo cual nos muestra que el hecho de añadir todas las características al sistema es perjudicial y un análisis previo es necesario. También observamos que el hecho de añadir las 4 mejores características parece proporcionar la mayor parte de toda la ganancia posible (que se consigue con las 10 mejores características).

Por tanto, esto nos viene a decir que toda la información se concentra en un porcentaje pequeño de características y que el resto, una vez añadidas las mejores, no aportan nada adicional.

Teóricamente, y si tuviésemos infinitos datos de entrenamiento, el hecho de añadir características no relevantes no debería empeorar el rendimiento del sistema, sino mantenerlo. Sin embargo, debido a que el conjunto de datos que se maneja es siempre finito, aunque suficientemente grande, el hecho de añadir características no relevantes, siempre puede empeorar algo el sistema, como aquí ocurre.

5.3.2 AMI

Hemos realizado los mismos experimentos, descritos anteriormente para ALBAYZIN, pero en este caso contra la base de datos de AMI.

5.3.2.1 Análisis Incremental de la Varianza

En las siguientes tablas mostramos el análisis incremental de la varianza para los conjuntos de datos que hemos analizado:

Analizing TRAIN balanced data feature that maximices R2		
FEATURE	NAME	R2
1	score	0.330557
3	effective_FA_rate	0.426483
6	number_phones	0.451832
31	min_duration	0.464197
27	max_languagemodelscore	0.467393
24	max_acoustic_score_normalizedbyphoneduration	0.469092
2	effective_hit_rate	0.470404
22	mean_energy	0.471643
21	min_energy	0.473357
37	min_energy_phoneme	0.474670
34	min_pitch_phoneme	0.475856
20	max_energy	0.477007
4	duration	0.478497
30	max_duration	0.481845
15	min_Levenshtein_distance	0.482785
33	max_pitch_phoneme	0.483280
14	max_Levenshtein_distance	0.483695
5	number_graphemes	0.484094
16	mean_Levenshtein_distance	0.484325
38	mean_energy_phoneme	0.484541
40	min_energy_phoneme_nonzero	0.485450
36	max_energy_phoneme	0.486239
25	min_acoustic_score_normalizedbyphoneduration	0.486474
39	min_pitch_phoneme_nonzero	0.486674
35	mean_pitch_phoneme	0.487143
28	min_languagemodelscore	0.487310
29	mean_languagemodelscore	0.487577
12	speaker_vowel_rate	0.487614
17	max_pitch	0.487644
32	mean_duration	0.487672
11	speaker_phone_rate	0.487767
19	mean_pitch	0.487789
18	min_pitch	0.487839

Analizing TRAIN balanced data feature that maximices R2		
FEATURE	NAME	R2
13	position	0.487850
23	voicing_percentage	0.487858
7	number_vowels	0.487863
8	number_consonants	0.487863
9	number_phone_vowels	0.487863
10	number_phone_consonants	0.487863
26	mean_acoustic_score_normalized_byphoneduration	0.487863

Tabla 8. Análisis Incremental de la Varianza para el conjunto de datos TRAIN de AMI

Analizing DEV prosodic balanced data feature that maximices R2		
FEATURE	NAME	R2
3	effective_FA_rate	0.566664
6	number_phones	0.637981
1	score	0.689927
15	min_Levenshtein_distance	0.701537
14	max_Levenshtein_distance	0.707446
8	number_consonants	0.715192
29	mean_languagemodelscore	0.718302
18	min_pitch	0.721268
33	max_pitch_phoneme	0.724876
9	number_phone_vowels	0.727518
16	mean_Levenshtein_distance	0.731034
7	number_vowels	0.739028
31	min_duration	0.742979
21	min_energy	0.745378
22	mean_energy	0.749081
23	voicing_percentage	0.751554
34	min_pitch_phoneme	0.753731
12	speaker_vowel_rate	0.755413
4	duration	0.757208
30	max_duration	0.758950
35	mean_pitch_phoneme	0.759893
39	min_pitch_phoneme_nonzero	0.763027
19	mean_pitch	0.764452
2	effective_hit_rate	0.765148
24	max_acoustic_score_normalizedbyphoneduration	0.765727
27	max_languagemodelscore	0.766252
28	min_languagemodelscore	0.767443
17	max_pitch	0.767673
20	max_energy	0.767790
26	mean_acoustic_score_normalized_byphoneduration	0.767908
25	min_acoustic_score_normalizedbyphoneduration	0.773522
32	mean_duration	0.773838
11	speaker_phone_rate	0.774619
37	min_energy_phoneme	0.774665
36	max_energy_phoneme	0.774687
40	min_energy_phoneme_nonzero	0.775725
38	mean_energy_phoneme	0.775777
5	number_graphemes	0.775792
10	number_phone_consonants	0.775792
13	position	0.775792

Tabla 9. Análisis Incremental de la Varianza para el conjunto de datos DEV de AMI

Analizing TEST.INV prosodic balanced data feature that maximices R2		
FEATURE	NAME	R2
1	score	0.340635
3	effective_FA_rate	0.434098
6	number_phones	0.458414
31	min_duration	0.468848
37	min_energy_phoneme	0.472982
21	min_energy	0.475339
28	min_languagemodelscore	0.477642
24	max_acoustic_score_normalizedbyphoneduration	0.479037
13	position	0.479736
9	number_phone_vowels	0.480123
8	number_consonants	0.480695
16	mean_Levenshtein_distance	0.481561
4	duration	0.481850
17	max_pitch	0.482148
34	min_pitch_phoneme	0.482384
26	mean_acoustic_score_normalized_byphoneduration	0.482510
25	min_acoustic_score_normalizedbyphoneduration	0.482706
38	mean_energy_phoneme	0.482808
29	mean_languagemodelscore	0.482889
27	max_languagemodelscore	0.483039
7	number_vowels	0.483110
12	speaker_vowel_rate	0.483216
32	mean_duration	0.483444
30	max_duration	0.483609
11	speaker_phone_rate	0.484039
14	max_Levenshtein_distance	0.484136
2	effective_hit_rate	0.484217
39	min_pitch_phoneme_nonzero	0.484280
18	min_pitch	0.484540
35	mean_pitch_phoneme	0.484622
33	max_pitch_phoneme	0.484794
19	mean_pitch	0.484881
23	voicing_percentage	0.484910
20	max_energy	0.484927
15	min_Levenshtein_distance	0.484932
36	max_energy_phoneme	0.484934
5	number_graphemes	0.484935
10	number_phone_consonants	0.484935
22	mean_energy	0.484935
40	min_energy_phoneme_nonzero	0.484935

Tabla 10. Análisis Incremental de la Varianza para el conjunto de datos TEST.INV de AMI

Analizing TEST.OOV prosodic balanced data feature that maximices R2		
FEATURE	NAME	R2
6	number_phones	0.626391
3	effective_FA_rate	0.665075
1	score	0.684451
16	mean_Levenshtein_distance	0.703204
5	number_graphemes	0.748295
29	mean_languagemodelscore	0.756377
14	max_Levenshtein_distance	0.761472
31	min_duration	0.764367
4	duration	0.764821
32	mean_duration	0.767894
30	max_duration	0.768823
12	speaker_vowel_rate	0.769330
23	voicing_percentage	0.769593
36	max_energy_phoneme	0.770312
21	min_energy	0.770781
22	mean_energy	0.771145
20	max_energy	0.771642
8	number_consonants	0.771962
38	mean_energy_phoneme	0.772299
37	min_energy_phoneme	0.772724
15	min_Levenshtein_distance	0.772974
18	min_pitch	0.773119
17	max_pitch	0.773581
10	number_phone_consonants	0.773710
26	mean_acoustic_score_normalized_byphoneduration	0.773834
25	min_acoustic_score_normalizedbyphoneduration	0.773923
2	effective_hit_rate	0.773982
28	min_languagemodelscore	0.774047
13	position	0.774089
34	min_pitch_phoneme	0.774123
11	speaker_phone_rate	0.774160
33	max_pitch_phoneme	0.774193
35	mean_pitch_phoneme	0.774237
27	max_languagemodelscore	0.774264
39	min_pitch_phoneme_nonzero	0.774288
24	max_acoustic_score_normalizedbyphoneduration	0.774296
7	number_vowels	0.774297
9	number_phone_vowels	0.774297
19	mean_pitch	0.774297
40	min_energy_phoneme_nonzero	0.774297

Tabla 11. Análisis Incremental de la Varianza para el conjunto de datos TEST.OOV de AMI

Podemos observar de nuevo como las características relacionadas con el lattice proporcionan la mayor discriminación entre las dos clases (acierto y falsa alarma). Concentrándonos en el conjunto de TRAIN, que es el que nos proporciona la mayor información por estar compuesto de palabras INV y OOV, y tratarse del que posteriormente se usará como datos de entrenamiento para medir la clasificación del Árbol de decisión, podemos apreciar lo siguiente: al contrario que los datos de ALBAYZIN, las características basadas en la duración no son tan importantes y sí las léxicas, al ocupar "number_phones" la tercera posición (Ver Tabla 8).

Sin embargo, y como ocurría en el caso de ALBAYZIN, el resto de características léxicas, por su redundancia, quedan en las peores posiciones del ranking. En esta base de datos, al ser voz espontánea, las duraciones no dan tanta pista como en el anterior conjunto de datos de ALBAYZIN, ya que las palabras pueden ser normalmente pronunciadas con diferente duración, al contrario a voz leída, donde la voz es más "plana". Sin embargo, las características léxicas sí aportan más, debido a que palabras cortas (con menos fonemas) son más difíciles de ser reconocidas que palabras largas (con más fonemas).

La alta correlación que existe entre las diferentes características léxicas hace que, aún ocupando una de ellas posiciones superiores, el resto quede en posiciones inferiores del ranking, lo que nos viene a decir, que una vez añadida la "mejor" el resto no aporta ninguna información adicional. De nuevo, como ocurría en la base de datos ALBAYZIN, las características derivadas del pitch apenas aportan información a la hora de discriminar entre acierto y falsa alarma.

Interesante es recalcar el hecho de que para los datos de DEV y TEST.OOV, el score no aparece en la primera posición, cuando realmente es esta característica la única que nos da información sobre la confianza que se tiene en que la palabra sea un acierto. Esto es debido a la escasez de datos en los conjuntos compuestos únicamente por palabras OOV (DEV y TEST.OOV), lo cual causa que el análisis de varianza no sea capaz de aprender un modelo robusto ante esta escasez de datos, quedando otras características a priori "peores" mejor posicionadas en el ranking final.

Sin embargo, debido a esta escasez de datos, cualquier conclusión de estos conjuntos puede llevar a equívocos y de ahí que no se pueda derivar ninguna interesante.

5.3.2.2 Evaluación de Atributos utilizando WEKA

También utilizamos el Evaluador de Atributos "GainRatioAttributeEval para los datos de AMI.

GainR(Class, Attribute) = *(H(Class) - H(Class | Attribute)) / H(Attribute)* *(18)*

AMI_Train_Ranker_GainRatio_hitFABalanced		
Ranked attributes:		
0.10515	1	Score
0.07465	6	number_phones
0.07362	9	number_phone_vowels
0.06818	10	number_phone_consonants
0.05764	5	number_graphemes
0.05073	8	number_consonants
0.04998	4	duration
0.04426	3	effective_FA_rate
0.04328	29	mean_languagemodelscore
0.04248	15	min_Levenshtein_distance
0.0373	25	min_acoustic_score_normalizedbyphoneduration
0.03717	34	min_pitch_phoneme
0.03713	22	mean_energy
0.0363	26	mean_acoustic_score_normalized_byphoneduration
0.03586	7	number_vowels
0.0355	27	max_languagemodelscore
0.03512	24	max_acoustic_score_normalizedbyphoneduration
0.03487	16	mean_Levenshtein_distance
0.03435	2	effective_hit_rate
0.03267	35	mean_pitch_phoneme
0.03194	40	min_energy_phoneme_nonzero
0.0317	37	min_energy_phoneme
0.03012	19	mean_pitch
0.02878	33	max_pitch_phoneme
0.02785	30	max_duration
0.02772	11	speaker_phone_rate
0.02696	32	mean_duration
0.02515	12	speaker_vowel_rate
0.02293	38	mean_energy_phoneme
0.02034	39	min_pitch_phoneme_nonzero
0.01799	23	voicing_percentage
0.01718	36	max_energy_phoneme
0.01667	28	min_languagemodelscore
0.01644	18	min_pitch
0.01511	17	max_pitch
0.01452	21	min_energy
0.01242	20	max_energy
0.00784	14	max_Levenshtein_distance
0.00591	31	min_duration
0.0013	13	position

Tabla 12. Evaluación de Características GainRatio en conjunto de datos TRAIN (AMI)

En este caso, las características que de forma individual proporcionan una mayor discriminación son las relacionadas con la información del lattice y las léxicas.

Contrario al análisis incremental, en este caso, como se puede ver, las características léxicas quedan en la parte superior del ranking, lo cual nos constata de nuevo que éstas son bastante importantes de forma individual cuando se trata de voz espontánea pero que, una vez añadida la mejor, aplicando un mecanismo incremental, el resto no aporta información adicional. Características prosódicas, como son el pitch y la energía son las peores situadas en el ranking individual, ya que, por si mismas y analizadas de forma individual, no aportan información sobre la discriminación entre las dos clases.

5.3.2.3 Clasificación a través de Árbol de Decisión

Utilizamos también la herramienta WEKA para hacer una clasificación de los atributos (basados en la evaluación con GainRatio) a través de un Árbol de Decisión (proporcionándole el conjunto de datos TRAIN como datos de entrenamiento y los conjuntos TEST.OOV y TEST.INV como datos de test).

Podemos ver, bajo estas líneas, el resumen de los resultados obtenidos añadiendo progresivamente la característica/s con mayor valor en la Evaluación anterior:

TEST .INV

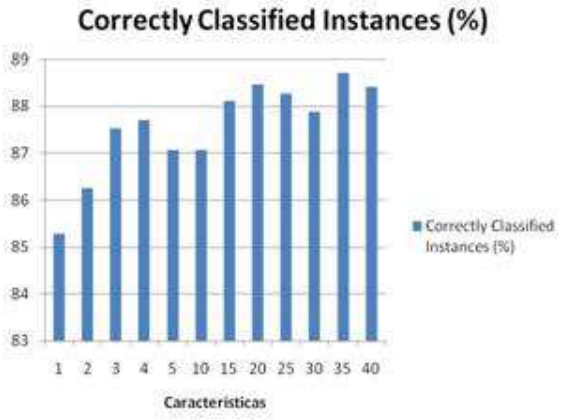


Figura 38. Porcentaje de ocurrencias correctamente clasificadas para el conjunto de datos TEST.INV (AMI)

TEST .OOV

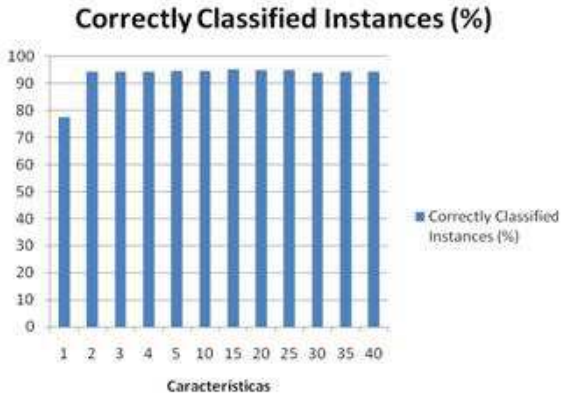


Figura 39. Porcentaje de ocurrencias correctamente clasificadas para el conjunto de datos TEST.OOV (AMI)

TEST .INV

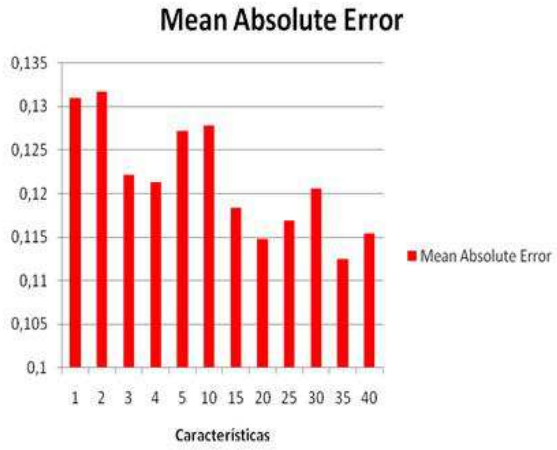


Figura 40. Tasa de Error Absoluto de Clasificación para el conjunto de TEST.INV (AMI). Nos da la medida de las diferencias en promedio entre los valores pronosticados y los observados

TEST .OOV

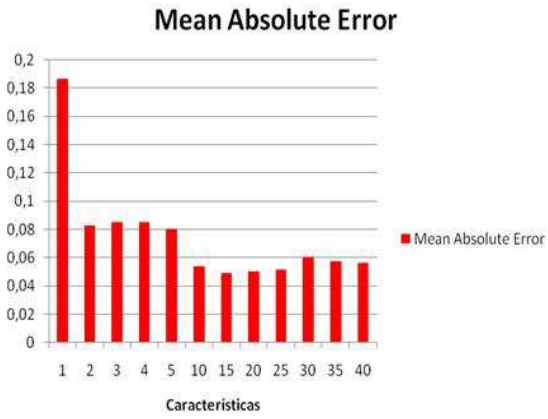


Figura 41. Tasa de Error Absoluto de Clasificación para el conjunto de TEST.OOV (AMI). Nos da la medida de las diferencias en promedio entre los valores pronosticados y los observados

De nuevo podemos observar como el uso de un conjunto reducido de características nos proporciona el mejor rendimiento posible a nuestro sistema en tareas de clasificación. En este caso, y teniendo en cuenta la dificultad de la base de datos que se ha tratado (voz espontánea), es necesario tener 20 características para obtener el mejor resultado para palabras INV, y entre 15 y 20 para palabras OOV.

De nuevo, el conjunto finito de datos que se maneja, hace que añadir características no relevantes pueda empeorar algo el sistema.

6

CONCLUSIONES Y TRABAJO FUTURO

ALBAYZIN

A través del análisis de la varianza explicada por cada una de las características, observamos que algunas de las nuevas introducidas en este proyecto explican un porcentaje muy alto de varianza (lo que da sentido a la reflexión inicial del proyecto, que nos llevó a pensar que sería interesante extraer estas nuevas características a nivel de fonema y que podrían mejorar el proceso de Decisión final dentro del Reconocedor de Voz).

Con el análisis incremental de la Varianza, generando un modelo completo con todas las variables añadiendo una a una la que más varianza residual explica, observamos que algunas de las nuevas características se colocan en las primeras posiciones. En concreto se coloca en 4ª posición la característica 31 (mínima duración por fonema), en 5ª la 26 (score acústico medio normalizado por duración del fonema) y en 7ª la 30 (máxima duración por fonema), y luego vienen la 25 (mínimo score acústico normalizado por duración del fonema) y la 36 (máxima energía de un fonema). (Analizando los datos de TRAIN).

También hay que destacar que ahora, con el modelo completo, para los datos de ALBAYZIN explicamos el 61.7% de la varianza (analizando los datos de TRAIN). En estudios anteriores (sin las nuevas características [13]) explicábamos sólo el 60.56%. Puede parecer poco un 1.14% más, pero es muy significativo porque cuando tienes muchas características subir un poco más el porcentaje de varianza explicada es muy importante.

Además también cabe destacar que existen bastantes características de diferentes grupos que quedan en las primeras posiciones, relativas a la confianza del lattice, léxicas, y de duración. Esto aporta una interesante conclusión, y es que las diferentes fuentes de información son muy importantes a la hora de establecer un ránking de características.

AMI

Es bastante razonable pensar que al trabajar ahora con voz más "difícil", el porcentaje de varianza explicada es menor que la base de datos de ALBAYZIN que es habla leída. Esto encaja perfectamente en la comparación entre las 2 bases de datos.

Por otro lado, lo que más llama la atención es la comparación entre el porcentaje de varianza explicada en los conjuntos DEV y TEST.OOV (donde son mucho mayores) frente a TRAIN y TEST.INV. Hay que tener en cuenta que en la base de datos de AMI, el conjunto de TRAIN está formado por palabras INV* y OOV**, mientras que el de DEV sólo por palabras OOV, por lo tanto, parece razonable pensar que el comportamiento de TRAIN y TEST.INV sea similar y que el de DEV y TEST.OOV (al estar formados ambos por palabras OOV) también sea similar.

Viendo los modelos formados añadiendo cada vez una característica, en TRAIN y TEST.INV el comportamiento es muy similar y bastante parecido a ALBAYZIN, aunque el porcentaje de varianza explicada total es menor.

Las palabras de DEV son OOV**, por lo tanto, sí que puede afectar como antes al hecho de que veamos similitudes entre DEV y TEST.OOV y totalmente diferente a TRAIN y TEST.INV. Por otro lado, hay que tener en cuenta cuando hablamos de OOV** en inglés, que difiere de INV* en que la transcripción fonética tiene errores, lo que es un factor muy a tener en cuenta cuando realizamos el análisis de estas características. Para ALBAYZIN sería algo más difícil de probar, puesto que el corpus geográfico apenas tiene palabras OOV, es decir, usando un conversor grafema-alófono la transcripción que se suele obtener casi siempre es la correcta.

Así que, a la vista de los resultados obtenidos, también podemos concluir que el comportamiento es claramente distinto para INV y para OOV, y la varianza explicada es sensiblemente mayor analizando los conjuntos con términos OOV, con el mismo grupo de características, lo cual es un resultado muy interesante científicamente ya que estos últimos son los términos generalmente más difíciles de tratar en el audio.

**inv:* palabras cuya transcripción fonética no tiene errores.
***oov:* palabras cuya transcripción fonética tiene errores.

De nuevo se puede ver también que diferentes fuentes de información contribuyen a un ranking de las características más preciso (podemos fijarnos, por ejemplo, en las 15 primeras características), si bien aquí las léxicas tienen mucha mayor importancia que en los datos de ALBAYZIN. AMI se trata de voz espontánea y es por ello que por ejemplo la duración esté peor clasificada que en ALBAYZIN, que es voz leída, por la espontaneidad de la primera, lo cual causa que la duración no sea un parámetro tan bueno como en voz leída.

ALBAYZIN

ALBAYZIN_Train_Ranker_GainRatio_hitFABalanced		
Ranked attributes		
0.10849	1	score
0.09807	24	max_acoustic_score_normalizedbyphoneduration
0.09234	27	max_languagemodelscore
0.08465	4	duration
0.08089	10	number_phone_consonants
0.08079	3	effective_FA_rate
0.07823	29	mean_languagemodelscore
0.07683	16	mean_Levenshtein_distance
0.07582	2	effective_hit_rate
0.07179	9	number_phone_vowels
0.07052	6	number_phones
0.06235	8	number_consonants
0.06082	28	min_languagemodelscore
0.06062	31	min_duration
0.05929	5	number_graphemes
0.05554	7	number_vowels
0.05419	12	speaker_vowel_rate
0.05221	11	speaker_phone_rate
0.0514	32	mean_duration
0.0426	30	max_duration
0.03952	25	min_acoustic_score_normalizedbyphoneduration
0.03806	26	mean_acoustic_score_normalized_byphoneduration
0.03165	22	mean_energy
0.03098	21	min_energy
0.02964	13	position
0.02524	23	voicing_percentage
0.02391	37	min_energy_phoneme
0.02343	40	min_energy_phoneme_nonzero
0.02322	14	max_Levenshtein_distance
0.01301	19	mean_pitch
0.01238	34	min_pitch_phoneme
0.01096	33	max_pitch_phoneme
0.01025	15	min_Levenshtein_distance
0.01009	38	mean_energy_phoneme
0.00994	17	max_pitch
0.00756	35	mean_pitch_phoneme
0.00746	18	min_pitch
0.00636	39	min_pitch_phoneme_nonzero
0.00567	36	max_energy_phoneme
0.00457	20	max_energy

AMI

AMI_Train_Ranker_GainRatio_hitFABalanced		
Ranked attributes:		
0.10515	1	Score
0.07465	6	number_phones
0.07362	9	number_phone_vowels
0.06818	10	number_phone_consonants
0.05764	5	number_graphemes
0.05073	8	number_consonants
0.04998	4	duration
0.04426	3	effective_FA_rate
0.04328	29	mean_languagemodelscore
0.04248	15	min_Levenshtein_distance
0.0373	25	min_acoustic_score_normalizedbyphoneduration
0.03717	34	min_pitch_phoneme
0.03713	22	mean_energy
0.0363	26	mean_acoustic_score_normalized_byphoneduration
0.03586	7	number_vowels
0.0355	27	max_languagemodelscore
0.03512	24	max_acoustic_score_normalizedbyphoneduration
0.03487	16	mean_Levenshtein_distance
0.03435	2	effective_hit_rate
0.03267	35	mean_pitch_phoneme
0.03194	40	min_energy_phoneme_nonzero
0.0317	37	min_energy_phoneme
0.03012	19	mean_pitch
0.02878	33	max_pitch_phoneme
0.02785	30	max_duration
0.02772	11	speaker_phone_rate
0.02696	32	mean_duration
0.02515	12	speaker_vowel_rate
0.02293	38	mean_energy_phoneme
0.02034	39	min_pitch_phoneme_nonzero
0.01799	23	voicing_percentage
0.01718	36	max_energy_phoneme
0.01667	28	min_languagemodelscore
0.01644	18	min_pitch
0.01511	17	max_pitch
0.01452	21	min_energy
0.01242	20	max_energy
0.00784	14	max_Levenshtein_distance
0.00591	31	min_duration
0.0013	13	position

Tabla 13. Comparativa entre ALBAYZIN y AMI de la Evaluación GainRatio

Como trabajo futuro se puede seguir investigando en el análisis y extracción de nuevas características, dado que se ha demostrado que utilizar diferentes fuentes de información ayuda a mejorar la clasificación de las mismas y por tanto a mejorar la detección final de un término. Podríamos hacer uso de otras herramientas como “Emovoice” (Análisis de Emociones), “Praat” (Análisis de entonaciones), etc.

Se podría continuar en la línea de este trabajo (análisis de características a nivel de fonema, aprovechando que hemos aislado la información de cada fonema que nos proporciona el lattice) y/o construir un “Decision Maker” para STD basándose en métodos discriminativos (MLP, SVM, CART) que calculen una probabilidad a posteriori para cada término detectado.

7

REFERENCIAS

[1] Javier Tejedor Noguerales, “Contributions to Keyword Spotting and Spoken Term Detection For Information Retrieval in Audio Mining”. Universidad Autónoma de Madrid. Departamento de Ingeniería Infomática.

[2] I. Szöke, P. Schwarz, P. Matejka, L. Burget, M. Karafiat, M. Fapso, y J. Cernocky. "Comparison of Keyword Spotting approaches for Informal Continuous Speech". Proc. of Interspeech, 2005, pages 633--636.

[3] NIST, “The Spoken Term Detection (STD) 2006 evaluation plan”, http://www.nist.gov/speech/tests/std/docs/std06-evalplan-v10.pdf, 2006.

[4] Javier Tejedor, Doroteo T.Toledano, Miguel Bautista, Simon King, Dong Wang and José Colás. “Augmented set of features for confidence estimation in spoken term detection”, InterSpeech 2010

[5] A. Moreno, D. Poch, A. Bonafonte, E. Lleida, J. Llisterri, J.B. Mariño, and C. Nadeu. “Albayzin speech database: Design of the phonetic corpus. In Proc. of European Conference on Speech Communication and Technology” (Eurospeech), volume 1, pages 653–656, September 1993.

[6] Carletta, J. (2006) Announcing the AMI Meeting Corpus. The ELRA Newsletter 11(1), January-March, p. 3-5.http://www.amiproject.org

[7] P. Boersma and D.Weenink, Praat: doing phonetics by computer, University of Amsterdam, Spuistraat 210, Amsterdam, Holland, 2007. [Online]. Available: http://www.fon.hum.uva.nl/praat/

[8] S. Young, G. Evermann, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland. The HTK Book (for HTK Version 3.2). Microsoft Corp. and Cambridge University Engineering Department, 2002.

[9] Dong Wang, "Out of Vocabulary Spoken Term Detection". http://www.era.lib.ed.ac.uk/handle/1842/4087

[10] School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. http://cmusphinx.sourceforge.net/

[11] School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. http://www.speech.cs.cmu.edu/

[12] R.E. De War and D.L. Neal. Weka machine learning project: Cow culling. Technical report, The University of Waikato, Computer Science Department, Hamilton, New Zealand, 1994. http://www.cs.waikato.ac.nz/ml/weka/

[13] D. Wang, S. King, J. Frankel, and P. Bell, "Term-dependent confidence for out-of-vocabulary term detection," in Proc. Interspeech' 09, Brighton, UK, September 2009, pp. 2139–2142.

[14] F. Wessel, K. Macherey, and R. Schlüter, "Using word probabilities as confidence measures," in Proc. ICASSP'98, vol. 1, Seattle, Washington, USA, May 1998, pp. 225 228.

[15] I. Szoke, P. Schwarz, P. Matejka, L. Burget, M. Karafiat, M. Fapso, and J. Cernocky. Comparison of keyword spotting approaches for informal continuous speech. In Proc. of International Conference on Spoken Language Processing (ICSLP), pages 633–636, September 2005.

[16] X. Huang, A. Acero, H. Hsiao-Wuen; "Spoken Language Processing", in Dr. R. Reddy, ed., New Jersey, Prentice Hall PTR, pp. 377-664, 2.001.

[17] Machine Intelligence Laboratory, Cambridge University Department of Engineering http://mi.eng.cam.ac.uk/mi/

[18] Luz García Martínez. "Ecualización de histogramas en el procesado robusto de voz" Universidad de Granada, departamento de teoría de la señal, telemática y comunicaciones.

[19] Cuayahuitl, H., Serridge, B.: Out-of-vocabulary word modelling and rejection for spanish keyword spotting systems. Proc. of MICAI, 155–165 (2002).

[20] Ou, J., Chen, C., Li, Z.: Hybrid neural-network/hmm approach for out-ofvocabulary words rejection in mandarin place name recognition. Proc. of ICONIP, (2001).

[21] Tejedor, J., García, R., Fernández, M., López-Colino, F., Perdrix, F., Macías, J.A., Gil, R.M., Oliva, M., Moya, D., Colás, J., Castells, P.: Ontology-based retrieval of human speech. Proc. of DEXA, 485–489 (2007).

[22] Setlur A R, Sukkar R A, Jacob J. Correcting recognition errors via discriminative utterance veri_cation. In Proc.ICSLP'96, Philadelphia, USA (October 1996) 602-605.

[23] Wallace R, Vogt R, Sridharan S. A phonetic search approach to the 2006 NIST spoken term detection evaluation. In Proc. Interspeech'07, Antwerp, Belgium (August 2007) 2385-2388.

[24] Gillick L, Ito Y, Young J. A probabilistic approach to confidence estimation and evaluation. In Proc. ICASSP'97, Munich, Bavaria, Germany (April 1997) 879-882.

8

GLOSARIO

ASR *(Automatic Speech Recognizer)*
Reconocimiento Automático de Habla.

CART *(Classification and Regression Tree)*
Proceso de construcción de modelos con estructura de árbol.

DET *(Detection Error Tradeoff)*
Las curvas DET se utilizan para la representación gráfica de las tasas de error de decisión (tasa de falso rechazo vs. tasa de falsa alarma).

HMM (*Hidden Markov Model*)
Modelo Oculto de Markov.

HTK (*Hidden Markov Model Toolkit*)
Conjunto de herramientas para la creación y tratamiento de HMMs, diseñado por la universidad de Cambridge.

Lattice
Representación compacta de una serie de hipotéticas posibles transcripciones para un archivo de audio concreto.

LVCSR *(Large Vocabulary Continous Speech Recognition)*
Sistema de reconocimiento de habla continua de vocabulario muy amplio.

MFCCs (*Mel-Frequency Cepstral Coefficients*)
Coeficientes cepstrales en escala de frecuencias Mel.

MLF (*Master Label Format*)
Formato de HTK para archivos que contienen transcripciones.

MLP *(Multi-Layer Perceptrons)*
Red neuronal multicapa.

NIST (*National Institute of Standars and Technology*)
Organismo federal, no regulador, perteneciente a la Cámara de Comercio de los Estados Unidos que desarrolla y promueve medidas, estándares y tecnología para aumentar la productividad, facilitar el comercio y mejorar la calidad de vida.

RAH
Reconocimiento Automática de Habla.

Sphinx
Conjunto de herramientas para el tratamiento de voz diseñado por la universidad de Carnegie-Mellon.

SPL *(Sound Pressure Level*)
Medida del nivel de presión de un sonido (se mide en decibelios).

STD *(Spoken Term Detection)*
Reconocimiento de palabras clave.

SVM *(Support Vector Machine)*
Conjunto de algoritmos de aprendizaje supervisado utilizado para problemas de clasificación, regresión y reconocimiento de patrones.

WEKA
Waikato Environment for Knowledge Analysis – (Entorno para Análisis del Conocimiento de la Universidad de Waikato) es una plataforma de software para aprendizaje automático y minería de datos escrito en Java y desarrollado en la Universidad de Waikato.

9

ANEXOS

A. Listado de Fonemas Españoles

La siguiente tabla muestra el listado de los fonemas utilizados en los experimentos sobre la base de datos en español. Junto a cada fonema, se muestra un ejemplo de palabra que lo contiene.

Fonema	Ejemplo	Fonema	Ejemplo
a	taz*a*	d	*d*uelo
e	mod*e*rador	D	verde
i	rub*i*o	f	rodol*f*o
o	cerr*o*	g	*g*aleón
u	j*u*gar	G	a*g*otar
A	t*a*za	X	hoja
E	c*e*rro	j	historia
I	lat*i*no	J/	*y*ogur
O	caraco*l*a	J	castilla*y*león
U	g*u*sto	k	*c*ama
an	alc*an*zar	l	ye*l*mo
en	conv*en*cer	L	capi*ll*a
in	d*in*ero	m	*m*ina
on	mo*on*taña	n	*n*ata
un	n*um*erar	Nn	gui*ñ*o
An	pl*an*ta	p	*p*arque
En	l*en*ta	r	amanece*r*
In	s*im*io	R	gue*rr*a
On	l*on*dres	s	*s*eto
Un	prof*un*do	t	*t*rabuco
b	a*b*ejorro	T	*z*amora
B	ca*b*allo	w	sig*üe*nza
T/	*ch*ico	gs	o*x*ígeno
N	hu*n*dir	-	-

Tabla 14. Listado de fonemas españoles junto con una palabra de ejemplo

.

B. Listado de Fonemas Ingleses

La siguiente tabla muestra el listado de los fonemas utilizados en los experimentos sobre la base de datos en inglés. Junto a cada fonema, se muestra un ejemplo de palabra que lo contiene.

Fonema	Ejemplo	Fonema	Ejemplo
hh	*h*urry	ow	c*o*ke
ey	oper*a*tor	p	*p*rint
aa	d*o*g	iy	quickl*y*
l	he*l*met	jh	anthropolo*g*y
ih	enr*i*ch	m	*m*an
s	fir*s*t	ao	*o*scillate
t	jus*t*ify	eh	optom*e*try
k	bro*k*en	dh	al*th*ough
ay	antiqu*a*ry	y	*y*umy
ax	*a*broad	uw	t*w*o
n	*n*ational	aw	*ow*lish
w	home*w*ork	sh	*sh*ake
b	a*b*ove	er	adapt*er*
ng	ri*ng*	g	*g*oat
ah	h*u*ng	uh	h*oo*ligan
f	*f*ly	th	*th*orn
v	*v*ein	eh	c*ha*rity
z	occupation*s*	oy	ann*oy*ing
d	indee*d*	zh	a*z*ure
ae	f*a*t	r	*r*eality

Tabla 15. Listado de fonemas ingleses junto con una palabra de ejemplo

C. Publicaciones en Congresos Internacionales

- Augmented set of features for confidence estimation in spoken term detection, publicado en InterSpeech 2010.

- Speech signal- and term-based feature contribution to hit/false alarm classification in a spoken term detection system, publicado en Fala 2010.

Augmented set of features for confidence estimation in spoken term detection

Javier Tejedor[1], *Doroteo T. Toledano*[2], *Miguel Bautista*[2], *Simon King*[3], *Dong Wang*[3] *and José Colás*[1]

[1]Human Computer Technology Laboratory,
[2] ATVS-Biometric Recognition Group,
Universidad Autónoma de Madrid, Spain
[3] The Centre for Speech Technology and Research,
University of Edinburgh, United Kingdom
javier.tejedor@uam.es

Abstract

Discriminative confidence estimation along with confidence normalisation have been shown to construct robust decision maker modules in spoken term detection (STD) systems. Discriminative confidence estimation, making use of term-dependent features, has been shown to improve the widely used lattice-based confidence estimation in STD. In this work, we augment the set of these term-dependent features and show a significant improvement in the STD performance both in terms of ATWV and DET curves in experiments conducted on a Spanish geographical corpus. This work also proposes a multiple lineal regression analysis to carry out the feature selection. Next, the most informative features derived from it are used within the discriminative confidence on the STD system.

Index Terms: confidence estimation, feature selection, spoken term detection, speech recognition

1. Introduction

Information retrieval from speech has received much interest in the last years, particularly for finding relevant information from audio archives. This led NIST to organise the Spoken Term Detection (STD) evaluation [1], and suggested the development of practical systems, including [2]–[9]. The standard STD architecture consists of an ASR subsystem to produce the word/sub-word lattices and a STD subsystem for term detection, as it is depicted in Figure 1.

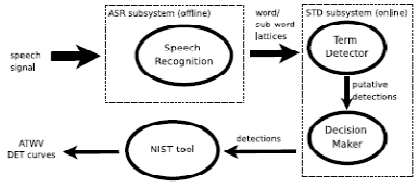


Figure 1: *The standard STD architecture: the speech recognition generates the lattices from the speech signal; a term detector searches these lattices for putative occurrences of the search terms; a decision maker sets if each putative detection is reliable. The NIST tool is used for system evaluation, in terms of ATWV and DET curves.*

Searching in the output of Large Vocabulary Continuous Speech Recognition (LVCSR) systems, i.e, word lattices, has been shown to work well when the query terms are just composed of in-vocabulary words. However, as claimed by Logan [10], at about 12% of the users' queries contain OOV words, which causes their search in word lattices cannot return any results. Common approaches to solve this problem consist of searching on sub-word lattices the phonetic representation of the enquiry terms ([11]–[13], among others). As this work focuses on OOV words, the STD system is built from a phone recognizer to extract a phone lattice and a term detection tool to search for putative occurrences of the enquiry terms through this lattice.

An essential component of a STD system is the *decision maker*, which examines each putative detection and decides if it is considered to be a hit or a false alarm (FA) based on confidence measures. In a previous work [14] the confidence for each detection $c_p(d)$ was derived from a mapping of three different lattice-based features:

$$g : (c_f(d), R_0(K), R_1(K)) \longrightarrow c_p(d) \quad (1)$$

where $c_f(d)$ is the lattice-based confidence proposed by Wessel et al. [15] and $R_0(K)$ and $R_1(K)$ represent the effective occurrence rate and the effective false alarm rate and are defined as follows:

$$R_0(K) = \frac{\sum_i c_f(d_i^K)}{T} \quad (2)$$

and

$$R_1(K) = \frac{\sum_i (1 - c_f(d_i^K))}{T} \quad (3)$$

where $c_f(d_i^K)$ represents the lattice-based confidence of the i-detection of the term K and T is the total length of the audio.

Next, to construct g, two well-known discriminative approaches (MLP and SVM) were employed, and the resulting confidence $c_p(d)$ was passed through a confidence normalisation process to compute the final confidence for each detection $\zeta(c_p(d))$. This discriminative confidence was shown to overcome the drawbacks of tradicional lattice-based confidence approaches [2],[5],[16]. Readers are referred to [14] for more details about the confidence normalisation. The mapping derived to construct g provides with a flexible framework in which multiple features can be easily integrated into the discriminative model to compose the discriminative confidence from a new mapping m as follows:

$$m : (c_f(d), R_0(K), R_1(K), f_0, f_1, ...) \longrightarrow c_p(d) \quad (4)$$

where f_0, f_1,... denote aditional features.

Next, the confidence normalisation converts this discriminative confidence $c_p(d)$ into $\zeta(c_p(d))$, which represents the final confidence for each detection d. Given the power of the discriminative confidence estimation for STD, we hypothesise in this work that the addition of new features in this mapping *may* enhance the hit/FA discrimination, leading to a significant improvement in the STD system. However, the choice of these aditional features is not an easy task and some quick and optimal mechanism is necessary since a random selection criteria is sub-optimal and may provide with worse performance. Therefore, the novelty of this work focuses on three different parts: 1) we present a putative set of relevant features for the new mapping m, 2) we conduct a lineal regression analysis to measure which features are more likely to contribute more to the discriminative confidence estimation and 3) we check the consistency of such analysis on STD performance. We chose the MLP as discriminative model since the previous work [14] did not report any meaningful difference between MLP and SVM.

The rest of the paper is organised as follows: The individual features used in the analysis and in the new mapping are described in Section 2. The experimental setup is presented in Section 3. The lineal regression-based feature analysis is presented in Section 4. The STD results are presented in Section 5. The work is discussed in Section 6 and concluded in Section 7.

2. Individual features

In building the mapping to the discriminative confidence estimator (MLP in our work) and partially inspired from the work presented by Goldwater et.al [17], where they studied a set of factors (features) that can contribute to a higher Word Error Rate (WER) in ASR systems, the following sets of features have been studied:

- Lattice-based features (LAT): This set of features comprises: the lattice-based confidence for each detection (i.e., $c_f(d_i^K)$, computed as in [14]), R0 (i.e, the effective occurrence rate for each term defined by Equation 2) and R1 (i.e, the effective false alarm rate for each term defined by Equation 3).
- Lexical features (LEX): This set of features comprises the total number of graphemes, vowel graphemes, consonant graphemes, phones, vowel phones and consonant phones for each term.
- Levenshtein distance features (LEV): The minimum, maximum and mean Levenshtein distance for each term against the rest.
- Duration features (DUR): This set of features comprises the duration of each detection, the duration divided by the number of phones (phone speech rate) and divided by the number of vowels (vowel speech rate) of each detection.
- Position (POS): It represents if the detection was found the first in the lattice, the last in the lattice or in any other position.
- Prosodic features (PROS): They comprise the pitch (maximum, minimum and mean pitch for each detection), the intensity (maximum, minimum and mean intensity for each detection) and the voicing percentage (i.e, the percentage of voiced speech for each detection in the speech signal). All these features were collected using Praat [18].

3. Experimental setup

The experiments were conducted on the geographical domain of the ALBAYZIN database [19]. The ***geographic training set*** was used for MLP training and parameter tuning while the ***geographic test set*** was used for the STD evaluation. We first selected 605 *OOV terms* from the ***geographic training set*** for MLP training and confidence normalisation. 500 terms of them, which had 12651 occurrences in this set (henceforth training set) were used for MLP training and 105 terms, with 10423 occurrences, (henceforth development set) for confidence normalisation tuning. For the STD evaluation, we selected 400 *OOV terms* which had 11331 ocurrences in the ***geographic test set*** (henceforth test set).

We built a phoneme-based speech recognizer from the HTK tool [20] in N-best mode to produce a phone lattice. It used state-clustered triphone models and 39-dimensional MFCC features. A bigram was used as LM trained from the ***phonetic training set***. A grapheme-to-phone conversor was used to predict pronunciations for all the *OOV terms*. As term detector, we used the *Lattice2Multigram* tool developed by Brno University of Technology (BUT).

We ran the STD system over the training set and detections were labeled as hit or false alarm prior to train the MLP whose parameters were optimised by cross-validation in this set. As in [14], to account for the imbalance between positive and training examples, we trained balanced models with the same number of hits and false alarms. As in [14] the confidence normalisation parameters were estimated by running the STD system over the development set. The STD evaluation, conducted over the test set, used the MLP trained from the training set and the confidence normalisation parameters tuned from the development set.

4. Lineal Regression Analysis

The main goal of our work is to improve confidence estimation for STD by expanding the set of features used in the previous work [14]. In Section 2 we have defined new sets of features that hopefully will improve the ability to estimate the confidence of a putative detection. Unfortunately, the computational cost of evaluating all combinations of these features with an MLP is prohibitive. Therefore we need to use a simpler and less costly method to find out the most interesting features and feature combinations to test within the MLP framework.

The method we have used is based on multiple linear regression. We start by balancing the number of hits and false alarms on the set used for MLP training. After that we apply multiple linear regression to explain the binary decision of classifying each putative detection as a false alarm (0) or as a hit (1) in terms of the features considered in Section 2. Instead of considering each feature individually, we considered them as belonging to the sets defined in Section 2. These sets could either be wholly included or wholly excluded from the multiple linear regression model. With these restrictions we performed a stepwise optimisation in which at each step the set of features that maximise the R^2 statistic was added to the model. This statistic can be interpreted as the amount of variance in the output variable that is explained by the multiple linear regression model, and the increment in R^2 can be interpreted as the additional variance explained with the introduction of the new set of features. Our hope is that the amount of additional variance explained by each set of features is related to the improvement achieved by the MLP when this set of features is added. Table

1 shows the results of this analysis, which indicate that Lattice-based features used in previous works [14] seem to be the most informative set of features, followed by duration, prosodic and lexical features. It is worth noting that the amount of additional variance explained by the additional set of features is dramatically reduced as new sets of features are added. This suggests that the amount of information added by the newly proposed features is somewhat residual, particularly for the two last sets of features added. In next section, we will obtain ATWV results using an MLP as confidence estimator using the sets of features in Table 1 to check for correlation between the multiple linear regression results and the MLP results.

Feature Sets	R^2 (%)	R^2 Increment (%)
LAT	52.3077	52.3077
+DUR	59.3618	7.0541
+PROS	60.0592	0.6974
+LEX	60.4241	0.3649
+LEV	60.5647	0.1406
+POS	60.5649	0.0002

Table 1: *Stepwise Multiple Linear Regression results. Feature sets are added in the order that maximises R^2. Results show the R^2 statistic in percentage and its absolute increment in percentage attributed to the last feature set added.*

5. STD results

5.1. Lattice-based features for discriminative confidence

We first present the improvement of the discriminative modelling over the lattice-based confidence for confidence estimation. Results presented in Table 2, and consistent with the previous work [14], show that the use of the discriminative confidence outperforms the lattice-based confidence. Moreover, in this work, the use of the discriminative confidence makes the ATWV possitive. As the terms chosen for the STD evaluation contain a number of phones that vary from 3 to 15, these 3-phone terms cause many FAs in the STD system, making necessary the discriminative confidence to achieve a positive ATWV. Pairwise t-tests show the significant improvement ($p < 0.001$) of the full set of lattice-based features over the single $c_f(d_i^K)$ feature in the discriminative confidence and over the lattice-based confidence. Therefore, for the augmented set of features for the discriminative confidence which is presented next, the full set of lattice-based features was selected as baseline.

5.2. Augmented features for discriminative confidence

Experiments used the different features presented in Section 2 for the MLP-based discriminative confidence estimation and results are presented in Table 3. These experiments were conducted in such a way that each set of feature is incrementally incorporated into the discriminative confidence as suggested by the multiple lineal regression analysis (see Section 4) (i.e, incorporating in each step the set of features that maximise R^2). Our results show that the new features proposed are able to improve significantly the STD performance. Pairwise t-tests show a significant improvement on the STD performance over the lattice-based features ($p < 0.001$) when the lattice, duration, and prosodic features are glued together to estimate the confidence and a weak significant improvement ($p < 0.03$) with these features over the combination of the lattice- and duration-based features. DET curves presented in Figure 2 present more

Confidence	Features	ATWV
Lattice-based	-	-0.034
Discriminative	$c_f(d_i^K)$	0.0617
Discriminative	$c_f(d_i^K)$, $R_0(K)$, $R_1(K)$	**0.2126**

Table 2: *STD performance with the lattice-based confidence and the discriminative confidence from the $c_f(d_i^K)$ and the full set of lattice-based features (i.e, $c_f(d_i^K)$, $R_O(K)$ and $R_1(K)$) with the best result in bold.*

remarkable differences for each set of features than a single operating point based on ATWV. They show that the augmented set of features that contribute a considerably increment in R^2 (above 0.6% in each step) outperform the lattice-based features either for all the range or for much of the range, specially for the set of features that maximise the STD performance in terms of ATWV. Contrary, the set of features that contribute marginally to such increment in each step (below 0.4%) achieves worse performance.

Discriminative confidence features	ATWV
LAT	0.2126
+DUR	0.2249
+PROS	**0.2379**
+LEX	0.2263
+LEV	0.2294
+POS	0.2284

Table 3: *STD performance according to the introduction of each set of features in the discriminative confidence with the best result in bold.*

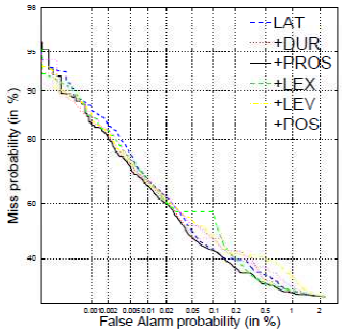


Figure 2: *The DET curves according to the relevance of the sets of features for discriminative confidence. The name of each curve is given as in Table 1.*

6. Discussion

By inspecting the multiple lineal regression analysis and STD performance measured from the discriminative confidence point of view, we can observe high correlation between both for most of the features. The lattice-based features, as they incorporate an actual *confidence* for each detection, explain much of the variance at first step in the lineal regression analysis. The duration was found to be added to the model in a second step.

This may be due short duration terms (may be caused by a fast speech) lead to more FAs. In addition, FAs can be produced by speech recognition errors that tend to produce awkward durations. Next, prosodic features provide with the highest increment in the variance. It may be due to, as stated in [17], terms detected in part of the speech with extreme values of pitch and energy contributed to more errors for ASR tasks (FAs in our case), causing these prosodic features can help detect them. The rest of the feature sets, which marginally contributed to an increment in R^2 (with a gain $\approx 0.6\%$ in total) lead to a worse STD performance. It may be due to the each individual contribution is absorbed by the features introduced in the previous steps. In addition, DET curves, which aims at giving a measure of how useful they are from different operating points show that they can dramatically reduce the overall STD performance for most of the range. This indicates that these non meaningful features affects negatively the mapping used to derive the final confidence for each detection. It must be also noted that R^2 is computed from the training data (i.e, the model fits the training data and the increment in the variance is computed from this set as well). Therefore, each new set of features added iteratively will never produce a reduction in R^2. Contrary, ATWV, is computed on the evaluation set from the model previously trained, and is affected by the generalisation level of the model (MLP in our case) in unseen data. This means that each partial increment in R^2 derived from the less informative features causes a worse STD performance, since they do not contribute to a better generalisation in the model.

7. Conclusions

This paper has investigated the introduction of relevant features in a discriminative confidence approach within the *decision maker* in a state-of-the-art STD system. It has been shown that lineal regression analysis can perform a robust feature selection to build the discriminative model mapping. Each aditional set of features is added to the discriminative model based on the increment in the variance explained from each. Experimental results also show that the addition of some of these new features to the lattice-derived features leads to a significant improvement on STD performance in terms of ATWV and DET curves. The features represented by different factors, such as prosodic, lexical, duration, position of each term found in the phone lattice and so on affect to a greater or lesser extent the performance levels of speech recognition systems. When the most informative features are integrated into a discriminative confidence estimation they may improve not only the STD performance as was investigated in this work, but also the performance of any general speech recognition system.

8. Acknowledgements

Part of this work was developed while JT was a visiting research at CSTR under the AMIDA Training Programme. This work also used the Edinburgh Compute and Data Facility which is partially supported by eDIKT. This work was also partially funded by the CAM S2009/TIC-1542 MA2VICMR project.

9. References

[1] NIST, *The spoken term detection (STD) 2006 evaluation plan*, 10th ed., National Institute of Standards and Technology (NIST), Gaithersburg, MD, USA, September 2006. [Online]. Available: http://www.nist.gov/speech/tests/std

[2] I. Szöke, M. Fapšo, M. Karafiát, L. Burget, F. Grézl, P. Schwarz, O. Glembek, P. Matějka, S. Kontár, and J. Černocký, "BUT system for NIST STD 2006 - English," in *Proc. NIST Spoken Term Detection Evaluation workshop (STD'06)*. Gaithersburg, Maryland, USA: National Institute of Standards and Technology, December 2006.

[3] D. R. H. Miller, M. Kleber, C. lin Kao, O. Kimball, T. Colthurst, S. A. Lowe, R. M. Schwartz, and H. Gish, "Rapid and accurate spoken term detection," in *Proc. Interspeech'07*, Antwerp, Belgium, August 2007, pp. 314–317.

[4] R. Wallace, R. Vogt, and S. Sridharan, "A phonetic search approach to the 2006 NIST spoken term detection evaluation," in *Proc. Interspeech'07*, Antwerp, Belgium, August 2007, pp. 2385–2388.

[5] D. Vergyri, I. Shafran, A. Stolcke, R. R. Gadde, M. Akbacak, B. Roark, and W. Wang, "The SRI/OGI 2006 spoken term detection system," in *Proc. Interspeech'07*, Antwerp, Belgium, August 2007, pp. 2393–2396.

[6] I. Szöke, L. Burget, J. Černocký, and M. Fapšo, "Sub-word modeling of out of vocabulary words in spoken term detection," in *Proc. IEEE Workshop on Spoken Language Technology (SLT'08)*, Goa, India, December 2008, pp. 273–276.

[7] S. Parlak and M. Saraçlar, "Spoken term detection for Turkish broadcast news," in *Proc. ICASSP'08*, Las Vegas, Nevada, USA, March 2008, pp. 5244–5247.

[8] C. Parada, A. Sethy, and B. Ramabhadran, "Balancing false alarms and hits in spoken term detection," in *Proc. ICASSP'10*, vol. 1, March 2010, pp. 5286–5289.

[9] D. Wang, S. King, and J. Frankel, "Stochastic pronunciation modelling for spoken term detection," in *Proc. Interspeech'09*, vol. 1, September 2009, pp. 2135–2138.

[10] B. Logan, P. Moreno, J.-M. V. Thong, and E. Whittaker, "An experimental study of an audio indexing system for the web," in *Proc. ICSLP'00*, vol. 2, October 2000, pp. 676–679.

[11] M. Saraclar and R. Sproat, "Lattice-based search for spoken utterance retrieval," in *Proc. HLT-NAACL 2004*, Boston, USA, May 2004, pp. 129–136.

[12] J. Mamou, B. Ramabhadran, and O. Siohan, "Vocabulary independent spoken term detection," in *Proc. ACM-SIGIR'07*, Amsterdam, The Netherlands, July 2007, pp. 615–622.

[13] D. Can, E. Cooper, A. Sethy, C. White, B. Ramabhadran, and M. Saraclar, "Effect of pronunciations on OOV queries in spoken term detection," in *Proc. ICASSP'09*, Taipei, Taiwan, April 2009, pp. 3957–3960.

[14] D. Wang, S. King, J. Frankel, and P. Bell, "Term-dependent confidence for out-of-vocabulary term detection," in *Proc. Interspeech'09*, Brighton, UK, September 2009, pp. 2139–2142.

[15] F. Wessel, K. Macherey, and R. Schlüter, "Using word probabilities as confidence measures," in *Proc. ICASSP'98*, vol. 1, Seattle, Washington, USA, May 1998, pp. 225–228.

[16] S. Meng, P. Yu, J. Liu, , and F. Seide, "Fusing multiple systems into a compact lattice index for Chinese spoken term detection," in *Proc. ICASSP'08*, Las Vegas, Nevada, USA, March 2008, pp. 4345–4348.

[17] S. Goldwater, D. Jurafsky, and C. D. Maning, "Which words are hard to recognize? prosodic, lexical, and disfluency factors that increase speech recognition error rates," *Speech Communication*, vol. 52, no. 3, pp. 181–200, 2009.

[18] P. Boersma and D. Weenink, *Praat: doing phonetics by computer*, University of Amsterdam, Spuistraat 210, Amsterdam, Holland, 2007. [Online]. Available: http://www.fon.hum.uva.nl/praat/

[19] A. Moreno, D. Poch, A. Bonafonte, E. Lleida, J. Llisterri, J. M. no, and C. Nadeu, "Albayzin speech database: Design of the phonetic corpus," in *Proc. Eurospeech*, September 1993, pp. 653–656.

[20] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *The HTK Book*, Engineering Department, Cambridge University, March 2006.

Speech signal- and term-based feature contribution to hit/false alarm classification in a spoken term detection system

Javier Tejedor[1], *Doroteo T. Toledano*[2], *Miguel Bautista*[2], *José Colás*[1]

[1]Human Computer Technology Laboratory,
[2] ATVS-Biometric Recognition Group,
Universidad Autónoma de Madrid, Spain
javier.tejedor@uam.es

Abstract

There are many factors that lead to decrease the final performance on spoken term detection (STD) systems. They are mainly related to the properties of the terms to be searched, the speech signal conditions and so on. This paper proposes and analyses a set of factors that can enhance or disminish the hit/false alarm (FA) ratio based on certain features. Our study reflects that detections corresponding to short-length terms, detections corresponding to a term similar to some other, short duration detections and lower confidence values assigned to each putative detection can lead to a FA whereas the opposite is shown to correspond to a hit in an open-vocabulary STD system.

Index Terms: spoken term detection, feature analysis, speech recognition.

1. Introduction

Speech information retrieval has received much interest for years, focusing on finding relevant information from audio archives. It encouraged many groups to develop practical systems [1–5] and NIST to conduct the first Spoken Term Detection (STD) evaluation [6], which aims at finding a list of terms fast and accurately in huge audio repositories. The standard STD architecture consists of a Speech Recogniser to produce word/sub-word lattices, a Term Detector to hypothesise putative detections and a Confidence Measure component to decide if each putative detection is reliable, as it is depicted in Figure 1.

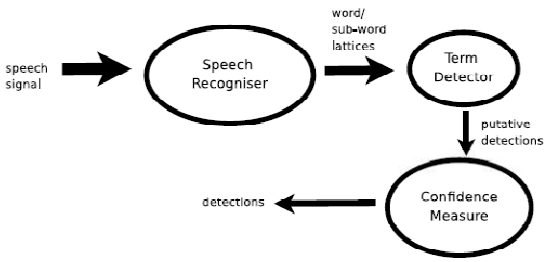


Figure 1: *The standard STD architecture.*

The *Confidence Measure* component plays a very important role in STD systems. It examines each putative detection and decides if it is considered to be a hit or a false alarm (FA). A *hit* occurs when a hypothesised detection appears in the speech signal. A *FA* occurs when the detection does not appear in the speech signal. An occurrence which is not hypothesised by the system is called a *miss*. Most of the works related to STD have proposed different confidence measures from which the final STD performance, in terms of ATWV and DET curves, is enhanced. Some are based on the scores produced by the speech recogniser [7,8]. Other such as n-best lists [9,10], minimum edit distance [11,12] and discriminative confidence [13–15] have been also explored. However, these works hardly make any analysis about which term properties or feature values derived from the speech signal are more likely to produce more hits or FAs. Actually, this hit/FA tradeoff measures the system performance. Therefore, this work aims at proposing a putative set of features, mainly term-based features, detection-based features and speech signal-based features and analyses their influence in the final hit/FA ratio. It must be noted that there are related works [16,17] which analyse the Word Error Rate (WER) contribution of individual words in an Automatic Speech Recognition (ASR) Large Vocabulary Continuous Speech Recognition (LVCSR) system. Our work is slightly different since we analyse the performance, in terms of hits and FAs in an open-vocabulary STD task. In addition, new features are also proposed and explored for this STD task.

The rest of the paper is organised as follows: Section 2 describes the sets of features explored in this work. Section 3 presents the experimental setup. An histogram-based analysis and linear regression-based analysis are presented in Section 4 and Section 5 respectively. Finally, the work is concluded in Section 6.

2. Feature class description

Inspired by the previous works [16,17], the following sets of features have been studied:

- Lattice features: This set of features comprises: the lattice-based confidence (score) for each detection (i.e., $c_f(d_i^K)$, computed as in [18] from standard forward-backward recursions), R0 (i.e., the effective occurrence rate for each term defined by Equation 1) and R1 (i.e., the effective false alarm rate for each term defined by Equation 2).

$$R_0(K) - \frac{\sum_i c_f(d_i^K)}{T} \quad (1)$$

$$R_1(K) = \frac{\sum_i (1 - c_f(d_i^K))}{T} \quad (2)$$

where $c_f(d_i^K)$ represents the lattice-based confidence of the i-detection of the term K and T is the total length of the audio.

- Lexical features: This set of features contains the total number of graphemes, phones, vowel graphemes, consonant graphemes, vowel phones and consonant phones for each term.
- Levenshtein distance features: The maximum, minimum and mean Levenshtein distance for each term against the others.
- Duration features: This set of features contains the duration of each detection, the duration divided by the number of phones (phone speech rate) and divided by the number of vowels (vowel speech rate) of each detection.
- Position: It represents if the detection appears the first in the lattice, the last in the lattice or in any other position.
- Prosodic features: They contain the pitch (maximum, minimum and mean pitch for each detection), the intensity (maximum, minimum and mean intensity for each detection) and the voicing percentage (i.e., the percentage of voiced speech for each detection in the speech signal). These features were collected using Praat [19].

The new features introduced in this work compared with the previous works [16,17] are the lattice-based features, all the lexical-based features except the number of phones, the Levenshtein distance features, the vowel speech rate within the duration features and the voicing percentage within the prosodic features.

3. Experimental setup

The geographical domain of the Albayzin database [20] was used for the experiments. 500 OOV terms, selected from the geographic corpus, which amount 12651 occurrences in the geographic training set, were used as list of terms. They were chosen based on their number of occurrences in this set.

A phone-based system was built from the HTK tool [21] in N-best mode to produce the phone lattices. It used state-clustered triphone models and 39-dimensional MFCC features. A bigram was used as LM trained from the phonetic training set of the Albayzin database. A grapheme-to-phone conversor was used to predict pronunciations for the *OOV terms*. As term detector, we used the *Lattice2Multigram* tool developed by Brno University of Technology (BUT), which hipothesises dectections based on an exact match of the phone transcription of each term and the paths in the phone lattice.

The STD system was run on the 500 OOV terms and the geographic training set and detections were labeled as hit or FA to carry out the analysis of which features are more likely to produce hits and FAs.

4. Histogram-based analysis

Each individual set of features explained in Section 2 is analysed from a histogram by plotting each feature contained in each group as it is presented in Figures 2-6. Inspecting the Figure 2, we see that, as expected, hits posses a higher score than FAs since it actually corresponds to the confidence assigned to each detection. Therefore, detections with higher scores are more likely to be hits and detections with lower scores should be considered as FAs. Consistent results are observed from the R0 and R1 features since terms with higher R0 and lower R1 are more likely to produce hits than FAs due to the former represents the effective occurrence rate and the latter represents the effective false alarm rate. Inspecting the Figure 3, where the lexical features per term are plotted, it can be seen that short-length terms (both in terms of phones are graphemes) are more likely to produce more FAs than long-length terms since the former can be a part of a long term or even a concatenation of the end and beginning of two different terms. This analysis is also consistent with the number of vowels and number of consonants (both for phones and graphemes). From the Figure 4, we observe that terms with a lower mean Levenshtein distance are more likely to be confused with some other and therefore they will produce more FAs than terms with higher mean Levenshtein distance, whose confusability with the rest is lower. However, extreme values (i.e., those derived from the maximum and minimum Levenshtein distances), does not separate hits from FAs in such a way that any clear conclusion is reached. Inspecting the duration-based features in Figure 5, we can see that a detection with a shorter duration is more likely to be a FA than a detection with longer duration, both in terms of absolute duration, phone speech rate and vowel speech rate. This may be due to many times FAs are produced by speech recognition errors that tend to cause awkward durations. The position of each detection found in the lattice does not discriminate between hits and FAs at all and therefore, the two plots are mostly overlapped in Figure 5. Finally, inspecting the Figure 6, we can see that detections corresponding to speech signal intervals with low intensity are more likely to be hits than FAs since higher values of minimum intensity may be caused by poor speech signal conditions and that the rest of the prosodic features do not discriminate between the hit and FA classes at all.

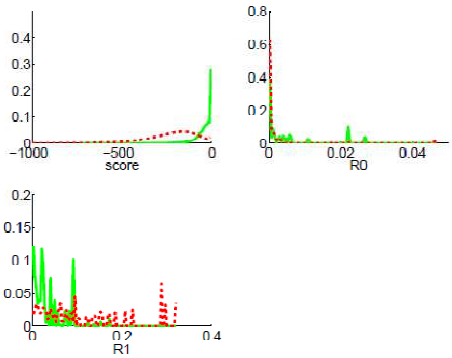


Figure 2: *Histogram analysis for the lattice-based features. Green bars represent hits and red bars represent false alarms.*

5. Linear regression-based analysis of variance

As an alternative analysis to the one presented in the former section, in this section we perform an analysis based on linear regression in which we analyse the amount of variance in the binary variable hit/FA, represented as a 1 or a 0, that can be explained by a linear regression using each of the individual features defined in Section 2. This analysis is performed using the stepwise function of MATLAB and computing the R^2 statistic.

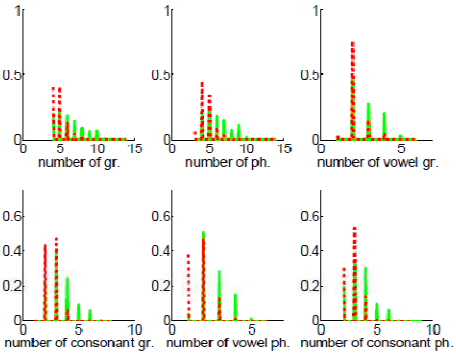


Figure 3: *Histogram analysis for the lexical features.* ph. *denotes phones and* gr. *denotes graphemes. The layout is the same as in Figure 2.*

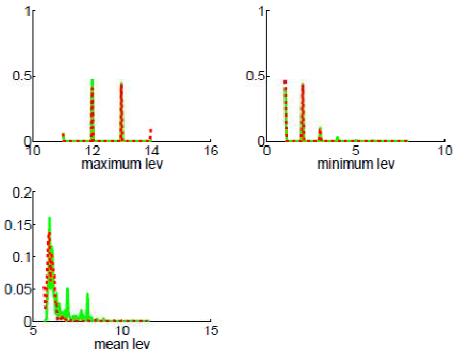


Figure 4: *Histogram analysis for the Levenshtein (lev) distance-based features. The layout is the same as in Figure 2.*

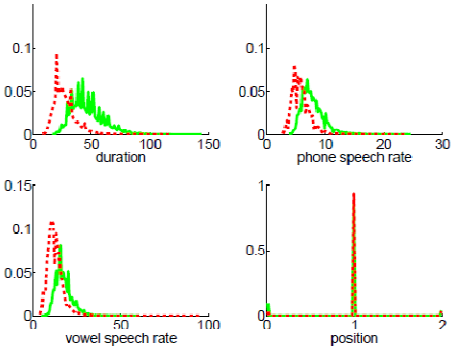


Figure 5: *Histogram analysis for the duration- and position-based features. The layout is the same as in Figure 2.*

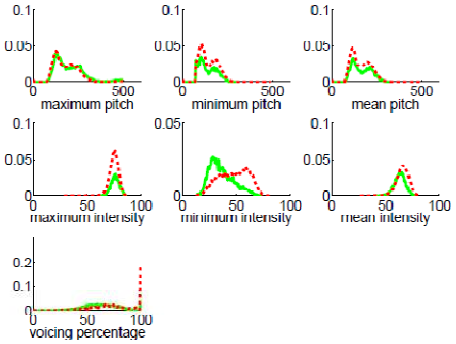


Figure 6: *Histogram analysis for the prosodic features. The layout is the same as in Figure 2.*

A similar approach was successfully used in [22] to choose the set of features that provides more information to discriminate between hits and FAs. There we showed that the conclusions obtained from the multiple linear regression analysis were in accordance with results obtained with a more complex (neural network) confidence estimator. Here our interest is different, because we are not interested in training a confidence estimator, but in determining the most interesting features in isolation. For this reason we do not group the features as we did there and only the percentage of reduction of variance achieved by using a single feature is analysed. Results of these analyses on the same set used in Section 4 are presented in Table 1.

This analysis yields basically the same conclusions obtained in the previous section, but with a numerical result that can be used to compare the amount of information provided by each individual feature in a more principled manner than by looking at the amount of overlapping of the histograms. Therefore, those feature histograms with a less overlapping between hit and FA classes lead to a higher R^2 contribution, which derives in a better hit/FA discrimination. Not surprisingly, the score is the feature that provides with the highest R^2, since it represents the confidence that the detection is considered to be a hit. It is consistent with the histogram-based analysis, where the score possesses the best hit/FA discrimination among all the features explored in this work. On the other hand, when the R^2 contribution of a certain feature is small, the histogram reveals a high degree of overlapping, meaning that such feature does not discriminate between both classes at all.

6. Conclusions

This work has investigated the individual contribution to the hit/FA classification in an STD system of both term- and detection-dependent properties and speech signal-based features. It has been shown that short terms are more likely to produce more errors and therefore more FAs in STD systems. Terms which posses a similar phone sequence are more likely to be confused with each other, leading to an increase in the FA rate, and short duration detections also contribute with a high FA rate.

Future work will investigate new features based on the most informative ones explored in this work, since it has been shown that lattice-, duration- and Levenshtein distance-based features

Feature Class	Feature	R^2 (%)
Lattice	score	42.48
Lattice	R0	4.06
Lattice	R1	20.29
Lexical	Number of graphemes	15.31
Lexical	Number of phones	18.81
Lexical	Number of vowel graphemes	10.97
Lexical	Number of consonant graphemes	13.15
Lexical	Number of vowel phones	20.58
Lexical	Number of consonant phones	8.59
Levenshtein distance	Maximum	1.23
Levenshtein distance	Minimum	1.02
Levenshtein distance	Mean	11.33
Duration	Duration of each detection	38.73
Duration	Phone speech rate	23.57
Duration	Vowel speech rate	21.84
Position	Position	0.98
Prosodic	Maximum Pitch	1.44
Prosodic	Minimum Pitch	0.25
Prosodic	Mean Pitch	0.10
Prosodic	Maximum Intensity	1.04
Prosodic	Minimum Intensity	17.12
Prosodic	Mean Intensity	5.64
Prosodic	Voicing percentage	4.49

Table 1: *Linear Regression analysis of variance results. Results show the R^2 statistic in percentage attributed to each feature, which can be interpreted as the percentage of variance explained by each particular feature.*

and lexical and prosodic features make an important contribution to hit/FA classification.

7. Acknowledgements

This work has been funded by the CAM S2009/TIC-1542 MA2VICMR project.

8. References

[1] I. Szöke, M. Fapšo, M. Karafiát, L. Burget, F. Grézl, P. Schwarz, O. Glembek, P. Matějka, S. Kontár, and J. Černocký, "BUT system for NIST STD 2006 - English," in *Proc. NIST Spoken Term Detection Evaluation workshop (STD'06)*. Gaithersburg, Maryland, USA: National Institute of Standards and Technology, December 2006.

[2] D. Vergyri, I. Shafran, A. Stolcke, R. R. Gadde, M. Akbacak, B. Roark, and W. Wang, "The SRI/OGI 2006 spoken term detection system," in *Proc. Interspeech'07*, Antwerp, Belgium, August 2007, pp. 2393–2396.

[3] S. Parlak and M. Saraçlar, "Spoken term detection for Turkish broadcast news," in *Proc. ICASSP'08*, Las Vegas, Nevada, USA, March 2008, pp. 5244–5247.

[4] C. Parada, A. Sethy, and B. Ramabhadran, "Balancing false alarms and hits in spoken term detection," in *Proc. ICASSP'10*, vol. 1, March 2010, pp. 5286–5289.

[5] J. Mamou and B. Ramabhadran, "Phonetic query expansion for spoken document retrieval," in *Proc. Interspeech'08*, Brisbane, Australia, September 2008, pp. 2106–2109.

[6] NIST, *The spoken term detection (STD) 2006 evaluation plan*, 10th ed., National Institute of Standards and Technology (NIST), Gaithersburg, MD, USA, September 2006. [Online]. Available: http://www.nist.gov/speech/tests/std

[7] J. R. Rohlicek, W. Russell, S. Roukos, and H. Gish, "Continuous hidden Markov modeling for speaker-independent word spotting," in *Proc. ICASSP'89*, Glasgow, UK, May 1989, pp. 627–630.

[8] S. Cox and R. Rose, "Confidence measures for the SWITCHBOARD database," in *Proc. ICASSP'96*, vol. 1, Atlanta, Georgia, USA, May 1996, pp. 511–514.

[9] M. Weintraub, "LVCSR log-likelihood ratio scoring for keyword spotting," in *Proc. ICASSP'95*, vol. 1, Detroit, Michigan, USA, May 1995, pp. 297–300.

[10] A. R. Setlur, R. A. Sukkar, and J. Jacob, "Correcting recognition errors via discriminative utterance verification," in *Proc.ICSLP'96*, Philadelphia, USA, October 1996, pp. 602–605.

[11] R. Wallace, R. Vogt, and S. Sridharan, "A phonetic search approach to the 2006 NIST spoken term detection evaluation," in *Proc. Interspeech'07*, Antwerp, Belgium, August 2007, pp. 2385–2388.

[12] K. Thambiratmann and S. Sridharan, "Rapid yet accurate speech indexing using dynamic match lattice spotting," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 1, pp. 346–357, January 2007.

[13] A. G. Hauptmann, R. E. Jones, K. Seymore, S. T. Slattery, M. J. Witbrock, and M. A. Siegler, "Experiments in information retrieval from spoken documents," in *Proc. DARPA Workshop on Broadcast News Transcription and Understanding*, Lansdowne VA, February 1998, pp. 175–181.

[14] K. Sudoh, H. Tsukada, and H. Isozaki, "Discriminative named entity recognition of speech data using speech recognition confidence," in *Proc. ICSLP'06*, Pittsburgh, USA, September 2006, pp. 1153–1156.

[15] Z. Shafran, B. Roark, and S. Fisher, "OGI spoken term detection system," in *Proc. NIST spoken term detection workshop (STD 2006)*, Gaithersburg, Maryland, USA, December 2006.

[16] S. Goldwater, D. Jurafsky, and C. D. Maning, "Which words are hard to recognize? prosodic, lexical, and disfluency factors that increase speech recognition error rates," *Speech Communication*, vol. 52, no. 3, pp. 181–200, 2009.

[17] S. Goldwater, D. Jurafsky, and C. D. Maning, "Which words are hard to recognize? lexical, prosodic, and disfluency factors that increase asr error rates," in *Proc. ACL/HLT*, June 2008, pp. 280–388.

[18] D. Wang, S. King, J. Frankel, and P. Bell, "Term-dependent confidence for out-of-vocabulary term detection," in *Proc. Interspeech'09*, Brighton, UK, September 2009, pp. 2139–2142.

[19] P. Boersma and D. Weenink, *Praat: doing phonetics by computer*, University of Amsterdam, Spuistraat 210, Amsterdam, Holland, 2007. [Online]. Available: http://www.fon.hum.uva.nl/praat/

[20] A. Moreno, D. Poch, A. Bonafonte, E. Lleida, J. Llisterri, J. M. no, and C. Nadeu, "Albayzin speech database: Design of the phonetic corpus," in *Proc. Eurospeech*, September 1993, pp. 653–656.

[21] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *The HTK Book*, Engineering Department, Cambridge University, March 2006.

[22] J. Tejedor, D. T. Toledano, M. Bautista, S. King, D. Wang, and J. Colás, "Augmented set of features for confidence estimation in spoken term detection," in *To appear in proc. Interspeech'10*, September 2010.

10

PRESUPUESTO

1) Ejecución Material
• Compra de ordenador personal (Software incluido)..... ..... 800€
• Material de oficina..... 50€
• Total de ejecución material..... **850**€

2) Gastos generales
• 21 % sobre Ejecución Material.....179 €

3) Beneficio Industrial
• 6 % sobre Ejecución Material.....….51 €

4) Honorarios Proyecto
• 920 horas a 15 € / hora..... ...13800 €

5) Material fungible
• Gastos de impresión..... .....150 €
• Encuadernación.....40 €

6) Subtotal del presupuesto
• Subtotal Presupuesto..... ….**15070** €

7) I.V.A. aplicable
• 21% Subtotal Presupuesto.....3165 €

8) Total presupuesto
• Total Presupuesto.....**18235** €

Madrid, Septiembre de 2012

El Ingeniero Jefe de Proyecto
Fdo.: Miguel Bautista Lozano

Ingeniero de Telecomunicación

PLIEGO DE CONDICIONES

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de un programa de búsqueda en lattices HTK con identificación y extracción de características fonéticas, prosódicas y de duración a nivel de fonema para la mejora en el reconocimiento de voz, y su posterior evaluación. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales

1. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15. La garantía definitiva será del 4% del presupuesto y la provisional del 2%.

16. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

23. Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad “Presupuesto de Ejecución de Contrata” y anteriormente llamado ”Presupuesto de Ejecución Material” que hoy designa otro concepto.

Condiciones particulares

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra

aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.